

MATLAB 工程仿真 与应用 30 例

施梨 编著

電子工業出版社

Publishing House of Electronics Industry

北京 · BEIJING

内 容 简 介

MATLAB 一个突出的特点是能够为工程实践提供强大有效的解决方案, 目前已在多个工程领域中得到广泛应用。本书着眼于工程实际, 一方面注重内容的实用性, 不仅详细介绍 MATLAB 工程应用的理论基础, 并且针对每个应用模块给出了典型的应用实例, 使读者在实际练习的过程中能快速提高应用水平; 另一方面结合工程应用的广泛性和集中性, 将全书分为控制、通信、电力电子、结构、热、图像和逻辑七大部分, 每一部分通过 4~6 个实例讲述 MATLAB 在某一个特定领域的工程应用, 从而使读者充分掌握 MATLAB 在多个工程领域的应用方法和应用过程。

随书提供案例源程序、教学视频等配套资源, 读者可登陆华信教育资源网(www.hxedu.com.cn) 搜索本书免费下载(须先注册)。

本书主要面向工科类的在校研究生和科研人员, 另外还可作为相关专业技术人员的参考用书。

未经许可, 不得以任何方式复制或抄袭本书之部分或全部内容。
版权所有, 侵权必究。

图书在版编目(CIP)数据

MATLAB 工程仿真与应用 30 例 / 施梨编著. —北京: 电子工业出版社, 2015.5

ISBN 978-7-121-25105-4

. M... . 施... . Matlab 软件 . TP317

中国版本图书馆 CIP 数据核字(2014)第 292696 号

策划编辑: 陈韦凯

责任编辑: 陈韦凯 特约编辑: 蒲 玥

印 刷:

装 订:

出版发行: 电子工业出版社

北京市海淀区万寿路 173 信箱 邮编: 100036

开 本: 787×1092 1/16 印张: 20.75 字数: 531 千字

版 次: 2015 年 5 月第 1 版

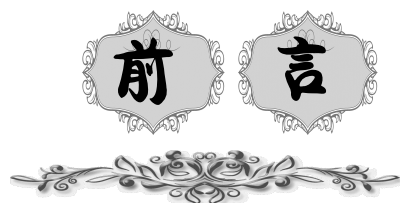
印 次: 2015 年 5 月第 1 次印刷

印 数: 3 000 册 定价: 54.00 元

凡所购买电子工业出版社图书有缺损问题, 请向购买书店调换。若书店售缺, 请与本社发行部联系, 联系及邮购电话: (010) 88254888。

质量投诉请发邮件至 zlts@phei.com.cn, 盗版侵权举报请发邮件至 dbqq@phei.com.cn。

服务热线: (010) 88258888。



MATLAB 是由美国 Mathworks 公司发布的主要面对科学计算、可视化及交互式程序设计的仿真计算环境。它将数值分析、矩阵计算、科学数据可视化及非线性动态系统的建模和仿真等诸多强大功能集成在一个易于使用的视窗环境中，为科学研究尤其是工程设计提供了一种全面的解决方案。本书精心挑选了 30 个工程仿真实例，用于展示 MATLAB 在工程仿真中的广泛应用和解决能力。

在编写本书时，我们努力遵循如下 5 点准则：

(1) 精心选择、安排实例内容。在工程实例选取时，主要选取船舶、飞机、汽车和卫星 4 个具有典型代表的工程对象。这 4 个工程对象系统复杂，代表了现代工程中多学科、多领域融合的方向。

(2) 难度由浅入深，易于理解。本书从 MATLAB 基础知识开始到各种工具箱介绍，使读者逐渐掌握 MATLAB 在工程实践中的多项应用。

(3) 涉及范围广、实例集中。书中介绍了 MATLAB 在控制、通信、电力电子、结构、热、图像和逻辑等多个工程领域的应用，但是实例主要集中在船舶、飞机、汽车和卫星 4 个工程对象，从多个方面展示工程对象的特征，更易于读者理解和掌握。

(4) 理论和实践相结合。书中的每一部分都是由理论基础到工程实际应用，读者既可以全面地了解理论知识，又可以掌握在工程中的使用方法。

(5) 目标和任务明确。每一个实例都介绍本例学习目标及本例小结，使读者在阅读时带着明确的任务，效率更高。

本书内容安排大致如下：

第一部分（第 1 例至第 4 例）为控制工程仿真，在介绍 MATLAB 编程基础和 Simulink 建模基础后，介绍基于 MATLAB 设计控制器的方法，并通过船舶、飞机、汽车和卫星 4 个实例予以介绍。

第二部分（第 5 例至第 8 例）为通信工程仿真，基础知识部分介绍了 MATLAB 通信工具箱、MATLAB 文件操作、串口操作和 S 函数第一部分。在实例部分以车载数字电视调制解调、舰载雷达通信系统、机载 GPS 信号接收及处理建模和 GPS 的 C/A 码及导航电文建模为例，介绍通信工程部分的仿真。

第三部分（第 9 例至第 12 例）为电力电子工程仿真，基础知识部分介绍了 SimPowerSystems 工具箱、Simdriveline 工具箱、RF 工具箱、Simscape 工具箱和 SimElectronics 工具箱基础知识。在实例部分介绍了燃料电池汽车仿真、雷达射频前端电

路仿真、飞机供电系统仿真和重力场卫星加速度计读取电路仿真。

第四部分（第 13 例至第 16 例）为结构工程仿真，基础知识部分介绍了 SimMechanics 工具箱和基于 M 语言的 GUI 设计。实例部分介绍了汽车 stewart 平台仿真、舰载四杆机构仿真、基于 SolidWorks 的 stewart 平台三维模型转换和卫星三维建模与有限元仿真。

第五部分（第 17 例至第 20 例）为热工程仿真，基础知识部分介绍了 Simscape 语言、Level-2S 函数和基于 C 语言 S 函数的使用方法。工程实例部分介绍了汽车温度调节系统仿真、船舶温度调节系统和卫星温度调节系统等仿真。

第六部分（第 21 例至第 24 例）为图像工程仿真，基础知识介绍了图像处理工具箱和地图工具箱，工程实例部分介绍了基于图像处理的交通车辆辨识、大型飞机航拍图处理、基于地图工具箱的船舶定位研究和卫星星下点轨迹图生成等仿真。

第七部分（第 25 例至第 30 例）为逻辑系统仿真，基础知识介绍了 Stateflow 工具箱基础知识，工程实例部分介绍了发射终止系统、月球登陆器自动驾驶仪、飞机俯仰轴容错控制、汽车电动车窗升降控制、汽车传动系统和导弹制导系统等仿真。

本书主要由施梨编著，此外，参与编写、修改工作的还有李龙、魏勇、王华、李辉、刘峰、徐浩、李建国、马建军、唐爱华、苏小平、朱丽云、马淑娟、周毅、张玉兰等。

本书编著者在编写过程中一直从读者的角度出发，力求通俗易懂，并充分考虑了当前工程实践的需求，其内容和难度符合广大学生和科研工作者在学习和生产实践中的使用需求。由于编著者水平有限，书中缺点和疏漏在所难免，恳请读者批评指正。

编著者

2015 年 2 月于上海松江

目 录



第一部分 控制工程仿真实例	1
引言——控制工程建模与分析方法	1
第 1 例 船舶运动控制仿真	7
1.1 MATLAB 编程基础	7
1.1.1 变量	7
1.1.2 运算符	11
1.1.3 常用数学函数	12
1.1.4 文件建立	15
1.2 船舶运动动力学及控制器	15
1.3 船舶运动控制器设计及仿真程序	16
1.4 本例小结	21
第 2 例 F-14 战斗机俯仰轴控制仿真	22
2.1 Simulink 建模及仿真基础	22
2.2 F-14 俯仰轴动力学模型	25
2.3 基于 Simulink 的 F-14 俯仰轴仿真模型	26
2.4 本例小结	29
第 3 例 汽车主动悬架控制器设计与仿真	30
3.1 汽车被动悬架系统仿真	30
3.1.1 被动悬架系统动力学模型	31
3.1.2 被动悬架系统 Simulink 模型	32
3.2 汽车主动悬架系统控制器设计	33
3.2.1 主动悬架系统动力学模型	33
3.2.2 主动悬架系统控制器设计及建模仿真	34
3.3 本例小结	37
第 4 例 卫星对地定向姿态控制设计	38
4.1 LMI 工具箱简介	38
4.1.1 LMI 基本概念	38



4.1.2	LMI 求解问题类型	39
4.1.3	LMI 建模求解函数	40
4.2	卫星对地定向动力学模型	42
4.3	控制器设计及仿真	43
4.4	本例小结	45
第二部分 通信工程仿真实例		46
引言——通信系统分类及 MATLAB 通信工具箱简介（上）		46
第 5 例 车载数字电视调制解调设计		56
5.1	MATLAB 通信工具箱简介（下）	56
5.2	车载数字电视调制解调设计	59
5.3	本例小结	63
第 6 例 舰载雷达通信系统仿真		64
6.1	S 函数简介	64
6.2	舰载雷达通信系统建模仿真	67
6.3	本例小结	71
第 7 例 机载 GPS 信号接收及处理过程建模		72
7.1	基于 MATLAB 文件操作简介	72
7.1.1	文件的打开与关闭	72
7.1.2	二进制文件的读/写操作	73
7.1.3	文本文件的读/写操作	74
7.1.4	MATLAB 读 txt 文件	74
7.2	机载 GPS 信号接收及处理建模	75
7.3	本例小结	80
第 8 例 GPS 卫星发射信号模拟		81
8.1	MATLAB 串口操作简介	81
8.2	GPS 的 C/A 码及导航电文建模	83
8.3	本例小结	89
第三部分 电力电子仿真实例		90
引言——SimPowerSystems 简介		90
第 9 例 燃料电池汽车动力系统仿真		96
9.1	Simdriveline 简介	96
9.1.1	SimDriveline 功能概述	96
9.1.2	SimDriveline 工具箱分类	98

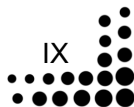
9.1.3 基于 SimDriveline 建模特点	99
9.2 燃料电池汽车仿真	99
9.2.1 燃料电池汽车简介	99
9.2.2 燃料电池汽车仿真电路设计	99
9.2.3 仿真结果及分析	102
9.3 本例小结	103
第 10 例 船舶雷达系统射频前端电路分析	104
10.1 RF 工具箱简介	104
10.1.1 基于 M 语言的 RF 工具箱特点及仿真过程	104
10.1.2 基于 Simulink 的 RF 工具箱分类	108
10.2 雷达射频前端电路设计与仿真	111
10.3 本例小结	112
第 11 例 飞机供配电系统设计与仿真	113
11.1 Simscape 工具箱简介	113
11.1.1 Simscape 功能及特点	113
11.1.2 Simscape 分类	114
11.1.3 Simscape 数学方程及仿真流程	115
11.2 飞机供配电系统建模与仿真	117
11.2.1 飞机供配电系统电路设计	118
11.2.2 仿真结果	121
11.3 本例小结	122
第 12 例 重力场卫星加速度计读取电路设计	123
12.1 SimElectronics 工具箱简介	123
12.1.1 SimElectronics 工具箱特点	123
12.1.2 SimElectronics 工具箱分类介绍	124
12.2 重力场卫星加速度计敏感电路设计与仿真	127
12.2.1 重力场卫星读取电路设计	127
12.2.2 仿真结果及分析	128
12.3 本例小结	131
第四部分 结构工程仿真实例	132
引言——SimMechanics 工具箱简介 (上)	132
第 13 例 车载 stewart 平台建模与仿真	141
13.1 SimMechanics 工具箱简介 (中)	141
13.2 车载 stewart 平台建模与仿真	145



13.3 本例小结	148
第 14 例 舰载雷达四杆机构建模与仿真	149
14.1 SimMechanics 工具箱简介 (下)	149
14.1.1 SimMechanics 工具箱可视化准备工作	149
14.1.2 可视化仿真窗口介绍	150
14.2 舰载雷达四杆机构仿真	152
14.3 本例小结	154
第 15 例 基于 SolidWorks 的机载 stewart 平台建模与仿真	155
15.1 从 CAD 建模工具中输入模型	155
15.1.1 转换步骤	155
15.1.2 生成模型特性	158
15.1.3 转换后模型修改	159
15.2 基于 SolidWorks 的 stewart 平台三维模型转换	160
15.3 本例小结	162
第 16 例 卫星三维建模与有限元分析	163
16.1 基于 M 语言的 GUI 界面设计	163
16.2 卫星三维建模与有限元分析	167
16.3 本例小结	173
第五部分 热工程仿真实例	174
引言——Simscape 语言简介 (上)	174
第 17 例 车载 stewart 平台建模与仿真	181
17.1 Simscape 语言简介 (中)	181
17.2 汽车温度调节系统仿真	186
17.3 本例小结	189
第 18 例 机载简易温度检测系统	190
18.1 Simscape 语言简介 (下)	190
18.2 机载简易温度检测仿真	194
18.3 本例小结	197
第 19 例 船舶温度调节系统	198
19.1 Level-2 S 函数简介	198
19.1.1 Level-2 S 函数基本特性	198
19.1.2 Level-2 S 函数模板	199
19.2 船舶温度调节系统	202

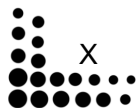


19.3 本例小结	204
第 20 例 卫星温度调节系统	205
20.1 基于 C 语言的 S 函数简介	205
20.2 卫星温度调节系统建模与仿真	211
20.3 本例小结	213
第六部分 图像工程仿真实例	214
引言——图像处理工具箱介绍（上）	214
第 21 例 汽车图像识别	221
21.1 图像处理工具箱介绍（中）	221
21.1.1 图像合成	221
21.1.2 图像的空间变换	222
21.1.3 邻域和块处理	224
21.2 基于图像处理的交通车辆辨识	225
21.3 本例小结	229
第 22 例 飞机航拍图处理	230
22.1 图像处理工具箱介绍（下）	230
22.1.1 图像分析	230
22.1.2 图像配准	232
22.2 大型飞机航拍图处理	234
22.3 本例小结	241
第 23 例 船舶定位研究	242
23.1 地图工具箱介绍（上）	242
23.1.1 创建地图	242
23.1.2 地理计算	244
23.2 基于地图工具箱的船舶定位研究	247
23.2.1 地图工具箱用于导航基本方法	247
23.2.2 船舶最短路程规划实例	250
23.3 本例小结	253
第 24 例 卫星星下点轨迹仿真	254
24.1 地图工具箱介绍（下）	254
24.1.1 地图投影	254
24.1.2 创建和查看地图	257
24.2 卫星星下点轨迹图生成	261
24.3 本例小结	262



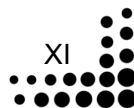


第七部分 逻辑系统仿真实例	263
引言——逻辑系统简介	263
第 25 例 发射终止系统仿真	264
25.1 Stateflow 状态模块与连接模块简介	264
25.1.1 状态模块	264
25.1.2 转移	267
25.2 发射终止系统	269
25.3 本例小结	272
第 26 例 月球登陆器自动驾驶仪仿真	273
26.1 Stateflow 其他模块的概念和基本用法	273
26.1.1 默认转移模块	274
26.1.2 历史节点	275
26.1.3 连接节点	276
26.1.4 盒子模块	278
26.1.5 连接分类	278
26.2 月球登陆器自动驾驶仪仿真	280
26.2.1 数据存储共享系统	281
26.2.2 动力学系统	281
26.2.3 开关逻辑生成系统	282
26.3 本例小结	283
第 27 例 飞机俯仰轴容错控制仿真	284
27.1 基于 Stateflow 建立有限状态机过程	284
27.1.1 建立 Stateflow Chart 内部结构	284
27.1.2 定义输入/输出变量	286
27.2 飞机俯仰轴容错控制仿真	288
27.3 本例小结	290
第 28 例 汽车电动车窗升降控制仿真	291
28.1 Stateflow 运行机理	291
28.1.1 有限状态自动机与 UML 状态图理论概述	291
28.1.2 Stateflow 机制分析与实现思路	293
28.2 汽车电动车窗升降控制仿真	295
28.2.1 指令输入部分	296
28.2.2 车窗动力学与控制部分	297
28.2.3 车窗控制逻辑部分	299
28.2.4 仿真结果	299





28.3	本例小结	300
第 29 例	汽车传动系统仿真	301
29.1	变速箱	302
29.1.1	行星齿轮组	302
29.1.2	离合器和制动带	303
29.2	引擎	304
29.3	液力变矩器	304
29.4	驱动系统及设备	306
29.5	发动机表格	307
29.6	变速逻辑	308
29.7	本例小结	309
第 30 例	导弹制导系统仿真	310
30.1	导弹三自由度动力学	310
30.1.1	三自由度导弹动力学	311
30.1.2	大气模型	313
30.1.3	自动驾驶仪模型	314
30.2	导弹制导系统	314
30.3	目标动力学	316
30.4	仿真结果	316
30.5	本例小结	317



第一部分 控制工程仿真实例



引言——控制工程建模与分析方法

控制工程是处理自动控制系统各种工程实现问题的综合性工程技术。包括对自动控制系统提出要求、进行设计、构造、运行、分析、检验等过程。

它普遍使用频域法和状态空间法。其理论和处理方法涉及多方面，从线性控制到非线性控制，从单变量控制到多变量控制，从连续系统控制到离散系统控制，从定常系统控制到随机系统控制等。控制工程的应用范围早期主要是工业生产过程，如化工、电子、冶金、电气、武器系统和火箭、卫星等，后来扩展到企业管理、城市规划、交通管制、生物控制、社会经济规划等领域。

第一部分选取船舶运动控制、F-14 俯仰轴控制、汽车悬架系统控制和卫星姿态控制为例介绍基于 MATLAB 如何对控制工程进行设计与仿真。其中 MATLAB 提供的控制工程工具箱能为控制工程仿真提供建模和分析提供有力的支持，因此在介绍实例之前首先对其进行介绍。

控制工程工具箱按照功能可分为两部分：建模和分析。

一、建模

对控制系统进行设计与仿真，首先需要建立控制系统的模型，控制工程工具箱提供了四种建模方式。

1. 状态方程形式

状态方程形式如式 (A1) 所示，该种形式在多变量线性系统中使用较多，便于控制器设计。

$$\begin{aligned}\frac{dx}{dt} &= Ax + Bu \\ y &= Cx + Du\end{aligned}\tag{A1}$$

式中 x 是状态变量， u 和 y 分别为输入和输出变量， A, B, C 和 D 具有适当的维数。

控制工程工具箱中建立状态方程的命令是 `ss`，下列程序给出一个建模实例。

```
R= 2.0; % Ohms  
L= 0.5 %; Henrys
```



```
Km = .015; % torque constant
Kb = .015; % emf constant
Kf = 0.2; % Nms
J= 0.02; % kg.m^2
A = [-R/L -Kb/L; Km/J -Kf/J];
B = [1/L; 0];
C = [0 1];
D = [0];
sys_dc = ss(A,B,C,D)
```

在 MATLAB 命令行输入上述程序（读者可在免费提供下载的配书资源中找到相应程序，下同）并运行后，得到：

```
a =
      x1      x2
x1      -4   -0.03
x2     0.75   -10

b =
      u1
x1      2
x2      0

c =
      x1  x2
y1      0   1

d =
      u1
y1      0
```

2. 传递函数形式

传递函数形如式（A2）所示。传递函数在单变量系统中使用较多，是经典控制主要采用的形式。

$$H(s) = \frac{s+2}{s^2+s+10} \quad (\text{A2})$$

式中 s 为拉普拉斯变换算子，关于其定义可参见复变函数相关书籍。

控制工程工具箱中建立状态方程的命令是 `tf`，在 MATLAB 命令行中输入：

```
sys_tf = tf(sys_dc) % sys_dc 是上一个程序建立的状态方程模型
```

即可得到：

```
Transfer function:
      1.5
-----
s^2 + 14 s + 40.02
```

3. 零极点增益形式

零极点增益形式如式（A3）所示。零极点增益形式便于了解系统的零极点分布，便于了解系统性能。



(A3)

$$H(z) = 3 \frac{(z+1+j)(z+1-j)}{(z+0.2)(z+0.1)}$$

控制工程工具箱中建立状态方程的命令是 `zpk`，在 MATLAB 命令行中输入：

```
sys_zpk = zpk(sys_dc) % sys_dc 是与一个程序采用相同的模型
```

即可得到：

```
Zero/pole/gain:
    1.5
-----
(s+4.004) (s+9.996)
```

4. 频域响应数据形式

该种方式通过对系统频域响应进行采样得到模型的信息。频域响应数据形式在模型未知或系统辨识中应用较多。如图 A1 所示，通过在系统 $G(\omega)$ 施加输入 $\sin \omega_i t$ ，测得对应的输出 $y_i(t)$ 并保存即得到频域相应的数据形式。

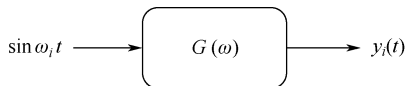


图 A1 频域响应测量方法

注意：四种形式可互换，如 `sys_tf = tf(sys_dc)` 即将状态方程形式转换为传递函数形式。

二、分析

控制工程工具箱提供了 LTI 观测器来进行对系统特性分析。LTI 观测器是一个观测线性系统响应及其他特性的 GUI 工具。LTI 观测器观测的对象为 LTI 对象，LTI 对象可利用前面介绍的建模工具来建立。通过 LTI 观测器可进行以下几方面工作。

- 阶跃响应、脉冲响应、初始状态响应和任意状态响应。
- 伯德图 (Bode)、奈奎斯特图 (Nyquist) 和 Nichols 图 (对数幅相图)。
- 频域相应的极点值。
- 系统零极点图。

通过在 MATLAB 命令行中输入 “`ltiview`” 即可打开 LTI 观测器，如图 A2 所示。基于 LTI 观测器分析系统性能可分为以下几步。

(1) 通过单击 “File|Import...” 即可打开 LTI 对象输入界面，如图 A3 所示。

(2) 在 LTI 对象输入界面中单击要分析的 LTI 对象，这里存在一个 LTI 对象 `sys_dc`，是之前在建模阶段建立的对象，然后单击 “OK” 即出现 `sys_dc` 阶跃响应图，如图 A4 所示。

(3) 在图 A4 中单击右键选择 “Characteristics|Rise Time” 可得到 `sys_dc` 阶跃响应上升时间，如图 A5 所示。

(4) 同样在图 A4 中单击右键选择 “Plot Types|Bode” 可得 `sys_dc` 伯德图，如图 A6 所示。

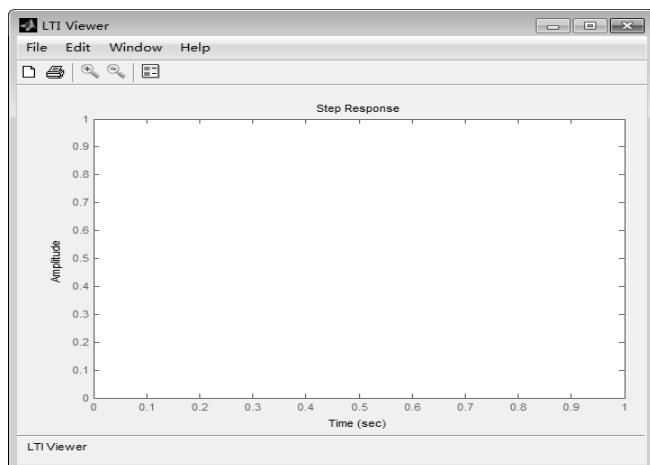


图 A2 LTI 观测器

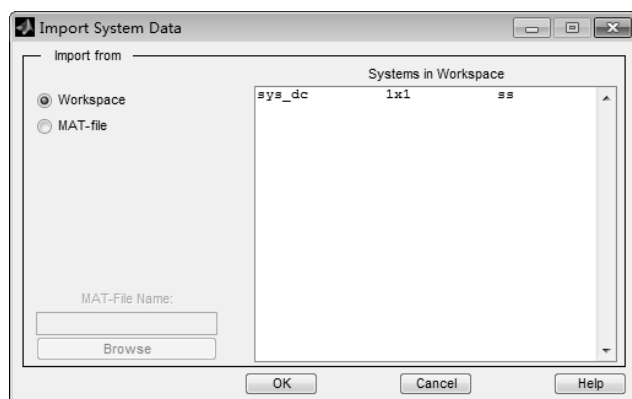


图 A3 LTI 对象输入界面

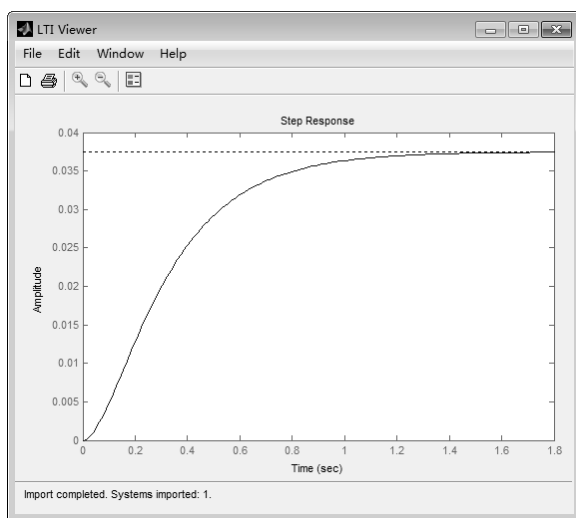


图 A4 sys_dc 阶跃响应图

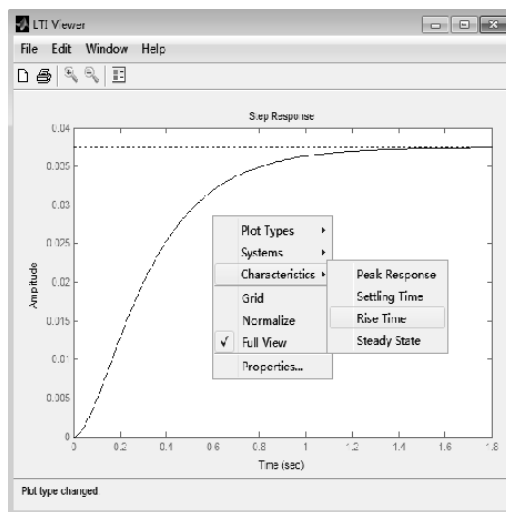


图 A5 sys_dc 阶跃响应上升时间

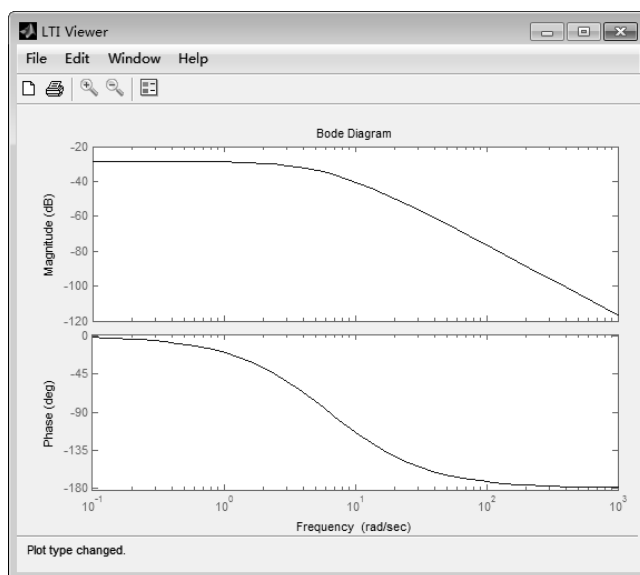


图 A6 sys_dc 伯德图

除 LTI 观测器外，MATLAB 控制工程工具箱还提供了相关函数，用来帮助对系统进行建模分析，表 A1 列出了这方面的部分函数。

表 A1 MATLAB 控制工程工具箱函数命令

函 数	说 明
frd	全 0 矩阵
frestimate	通过估计 Simulink 模型建立 frd 对象
ss	建立控制对象状态方程表达式



续表

函 数	说 明
tf	建立控制对象传递函数表达形式
zpk	建立控制对象零极点增益表达形式
get	得到 LTI 对象属性
set	设置 LTI 对象属性
step	系统阶跃响应
size	如果是 LTI 对象则返回输入输出个数，如果是矩阵则返回矩阵每一维的维数
ndims(model_name)	维数
isct(model_name)	如果是连续系统则返回 1
isdt(model_name)	如果是离散系统则返回 1
hasdelay(model_name)	如果系统有延迟则为真
pole(model_name)	系统极点
zero(model_name)	系统零点
dcgain(model_name)	系统增益
norm(model_name)	系统范数
covar(model_name,W)	系统白噪声相应协方差
bandwidth(model_name)	系统带宽
lti/order(model_name)	LTI 模型阶数
pzmap(model_name)	LTI 零点极点图
damp(model_name)	系统极点自然频率和阻尼
isempty(model_name)	判断对象是否为空
issiso(model_name)	判断对象是否为单输入单输出系统
lti/isstable(model_name)	判断系统是否稳定
sys1 + sys2	两个系统相加
sys1 - sys2	两个系统相减
sys1 * sys2	两个系统相乘
feedback	两个系统通过反馈的方式连接
c2d	连续系统转为离散系统
d2c	离散系统转为连续系统
ltiview	LTI 观测器
bode	伯德图
impulse	脉冲响应计算
initial	初始状态响应
lsim	对任意输入响应
nichols	nichols 图
nyquist	nyquist 图

注意：其中 ss、tf 和 zpk 也可以建立控制对象离散形式。



第 1 例 船舶运动控制仿真

船舶运动控制一直是国内外研究的热点。一方面，船舶运动控制系统表现出时变和非线性的特点；另一方面，控制性能要求节能、安全和强鲁棒性，控制目标由航向控制到航迹控制，以实现复杂环境下的自动航行和自动靠泊等。

本例基于 MATLAB 和采用 PD 算法对船舶运动设计控制器，并进行仿真。在设计之前，为使读者熟悉基于 MATLAB 语言的设计与仿真过程，首先对 MATLAB 编程基础进行简单介绍，其次介绍船舶运动动力学与控制器设计。

通过本例学习应了解和掌握以下几部分：

- MATLAB 编程基础。
- 船舶动力学基础。
- 船舶控制器。

1.1 MATLAB 编程基础

构成 MATLAB 编程的基本要素有变量、运算符、函数及文件四类。其中变量是构成 MATLAB 语言的最基本单位之一，用于表示自定义对象和特殊对象等；运算符用于连接变量，通过组合表示更为复杂的对象；函数具有输入变量和输出变量，通过一定的算法将输入变量与输出变量联系起来；文件是 MATLAB 保存程序和数据的载体，当程序较大时，需要通过一个或多个文件保存。下面对这四类要素分别予以介绍。

1.1.1 变量

变量可以表示程序中用到的数值和矩阵变量及字符变量等。为使用变量，首先需要利用标识符将变量表示出来。本小节分别介绍变量名标识符表示准则、矩阵建立和字符变量建立方法。

1. 变量名标识符表示准则

变量名利用标识符表示。标识符是标志变量名、常量名、函数名和文件名的字符串的总称。标识符表示变量等需要遵循以下一些准则：

- 变量和常量的标识符长度不超过 31（6.5 版本以后为 63）个字符。
- 标识符中的第一个字符必须是英文字母，不能有汉字。



- 标识符可以包含下划线、数字，但不能为空格符、标点。
- 大小写敏感。
- 变量无需事先定义即可使用。

变量名标识符例子：boat, car_tv, satellite1 等。

注意：标识符不能占用 MATLAB 专用常量及变量名，表 1-1 列出了 MATLAB 部分特殊变量和常数标识符。此外为便于理解，如果习惯用英文，则都用英文，如果习惯用拼音，则都用拼音。

表 1-1 MATLAB 部分特殊变量和常数

函 数	说 明
ans	最近生成的无名结果
eps	计算机的零阈值
pi	3.14159265358979
i	虚数单位
j	虚数单位
Inf	无穷大
NaN	Not-a-Number，例如 0/0
computer	计算机的类型
inputname	输入变量名

2. 矩阵变量建立

使用矩阵是 MATLAB 数值计算的一大特点，通过建立矩阵变量能采用一些基于矩阵的快捷数值算法，从而能有效地提高计算效率。建立矩阵分为直接输入法、步长生成法和部分输入法三种，此外如果矩阵元素为复数，也有相应的输入方法，而 MATLAB 自身也提供了部分特殊矩阵建立及矩阵信息查询的函数。下面对这五部分内容分别予以介绍。

1) 直接输入法

直接输入法例子：

在 MATLAB 命令行中输入 `a = [1 2 3; 4 5 6; 7 8 9]` 并回车，得到：

```
a =  
    1    2    3  
    4    5    6  
    7    8    9
```

可见 a 为一个三行三列的矩阵。

注意：矩阵的值放在方括号[]中，矩阵列间用空格或逗号“,”分隔，而行间用分号“;”分隔。另外输入时需英文输入格式下输入。

2) 步长生成法

步长生成法例子：

在 MATLAB 命令行中输入 `t = 0:1:10` 并回车，得到：



```
t =
    0     1     2     3     4     5     6     7     8     9    10
```

可见 t 为一个 1 行 11 列的矩阵（同时为向量）。

注意：步长生成法的格式为：初值:步长:终值，中间的数值为步长，可以为正也可以为负。如果为正，则终值需大于初值，反之则终值需小于初值。

3) 部分输入法

部分输入法例子：

在 MATLAB 命令行中输入 $a = [1\ 2\ 3; 4\ 5\ 6; 7\ 8\ 9]$ 、 $t = 0:1:10$ 、 $c = [a; t(1,1:3)]$ ，并回车，得到：

```
c =
     1     2     3
     4     5     6
     7     8     9
     0     1     2
```

可见 c 为一个 3 行 4 列的矩阵。而且可以看出 c 矩阵的前三行是由 a 得到，而最后一行是由 t 的前三个元素组成。这就说明可以在已有矩阵的基础上通过组合得到更为复杂的矩阵，从而能够提高效率。

提示：如果不想在命令行中看到变量的具体展开形式，则在命令的后面加上“;”即可，如“ $t = 0:1:10;$ ”后面加上“;”，则在命令行中就不会显示 t 的展开形式。

部分输入法中涉及矩阵下标的概念。如 $t(1,1:3)$ ，其中 t 括号里面的 1 以及 1:3 都为下标，前面的 1 表示第 1 行，后面的 1:3 表示第 1 列至第 3 列。利用下标，矩阵可以迅速地定位矩阵元素信息。如 $t(1,1:3)$ 即表示 t 矩阵中第 1 行中的第 1 列至第 3 列这三个元素，而 $t(1,1)$ 则表示 t 矩阵中第 1 行第 1 列这个元素。

提示：矩阵下标也可以运用步长生成格式，如 $t(1,1:2:5)$ 表示 t 矩阵中第 1 行中的第 1 列、第 3 列以及第 5 列这三个元素。

4) 复数矩阵建立

复数矩阵例子：

在 MATLAB 命令行中输入以下内容：

```
z1 = 8 + 10i;           % 直接按直角坐标的方式输入
z2 = 8 + 10*i;          % 运算符构成的直角坐标的方式输入
z3 = 25*exp(i*pi/3);    % 运算符构成的极坐标的方式输入
A = [z1,z2,z3] %矩阵形式
```

则可得到：

```
A =
 8.0000 +10.0000i   8.0000 +10.0000i  12.5000 +21.6506i
```

例子中存在三种复数输入方式：第一种是数值直接置于“ i ”左边，第二种是数值通过“ $*$ ”号与“ i ”连接起来，第三种是“ i ”在前面，通过“ $*$ ”号与数值连接。三种方式



都是可行，且具有相同的效果。

提示：在 MATLAB 中 “%” 后的内容为注释。

5) 常用与矩阵相关的函数命令

MATLAB 部分特殊矩阵建立及矩阵信息查询的函数命令，如表 1-2 所示。

表 1-2 MATLAB 部分特殊矩阵建立及矩阵信息查询的函数命令

函 数	说 明
zeros	全 0 矩阵
ones	全 1 矩阵
magic	魔方矩阵
rand	0~1 之间的均匀分布的伪随机数
randn	均值为 0，方差为 1 的正态分布
eye	单位矩阵
linspace	线性分布
logspace	对数分布
'	矩阵转置
fliplr	矩阵左右翻转
flipud	矩阵上下翻转
rot90	矩阵整体逆时针旋转 90 度
diag	提取或建立对角阵
tril	取矩阵的左下三角部分
triu	取矩阵的右上三角部分
reshape	维数重组（元素总数不变）
length	返回矩阵较大维数的数值
size	返回矩阵各维维数的数值

提示：函数的具体用法可通过 MATLAB 的帮助文档详细了解。

3. 字符变量建立

字符变量建立例子：

在 MATLAB 命令行中输入以下内容：

```
S1 = 'Any Characters'
S2 = 'All Characters'
S = [S1 S2]
C = {S1 S2}
```

则可得到：

```
S1 =
Any Characters
S2 =
All Characters
S =
Any CharactersAll Characters
```

```
C =  
    'Any Characters'    'All Characters'
```

由上面的例子可以看出：

- 通过两个单引号组合可建立字符变量。
- 字符变量也可以作为矩阵元素，即变量也可以表示字符矩阵。
- 大括号包含的整体为元胞矩阵，里面的元素为元胞。元胞是 MATLAB 中一种数据类型，可以为字符、数值等多种类型。详细内容可“ doc cell ”命令进行了解。

为便于字符变量的操作，MATLAB 提供了部分涉及字符变量操作的函数命令，如表 1-3 所示。

表 1-3 MATLAB 部分字符变量操作函数

函 数	说 明
strcat	将多个字符变量按水平方向连接起来
char（输入为多个字符变量）	将多个字符变量按垂直方向连接起来
char（输入为字符元胞数组）	将字符元胞矩阵变为字符数组
char（输入为数值变量）	将数值变量变为字符变量
double（输入为字符变量）	将字符变量变为数值变量
cellstr	将字符数组变为字符元胞矩阵
ischar	是否是字符
iscellstr	是否是元胞字符

1.1.2 运算符

运算符是 MATLAB 数值运算以及连接变量的基本符号 ,MATLAB 提供了丰富的运算符号，如表 1-4 所示。

表 1-4 MATLAB 常用运算符

运 算 符	说 明
+	加
-	减
*	乘
/	右除
\	左除
^	幂积
.*	点乘，同维矩阵对应元素做乘
./	点除，同维矩阵对应元素做右除
.\	点除，同维矩阵对应元素做左除
.^	点幂，对矩阵的每个元素求幂
<	小于，如果为真，返回 1，否则返回 0



续表

运 算 符	说 明
<=	小于等于，如果为真，返回 1，否则返回 0
>	大于，如果为真，返回 1，否则返回 0
>=	大于等于，如果为真，返回 1，否则返回 0
==	等于，如果为真，返回 1，否则返回 0
~=	不等于，如果为真，返回 1，否则返回 0
&	逻辑与，如果为真，返回 1，否则返回 0
	逻辑或，如果为真，返回 1，否则返回 0
~	逻辑非，如果为真，返回 1，否则返回 0
=	赋值
%	注释符号
:	冒号运算符

1.1.3 常用数学函数

除如表 1-1 至表 1-3 所列的函数外，MATLAB 还有一些常用的关于数值计算的函数，此外为将数值运算结果用图形表示出来，MATLAB 还提供了丰富的关于图形操作的函数，下面对这两方面函数予以介绍。

1. 常用数值函数

表 1-5 所示为 MATLAB 部分常用的数值函数，这些函数的具体用法可以在 MATLAB 命令行中输入“help 函数名”来查看。

表 1-5 MATLAB 部分常用数值函数

函 数	说 明
sin	正弦函数
cos	余弦函数
tan	正切函数
sinh	双曲正弦函数
cosh	双曲余弦函数
sech	双曲正割函数
asin	反正弦函数
acos	反余弦函数
atan	反正切函数
exp	以 e 为底的指数函数
power	$\text{power}(A,B) = A.^B$
sqrt	平方根函数
pow2	以 2 为底的指数

续表

函 数	说 明
nextpow2	返回 $2^N \geq x$ 的最小 N 值
log	对数函数
log10	以 10 为底的对数函数
log2	以 2 为底的对数函数
abs	绝对值
angle	辐角函数
real	取实部函数
imag	取虚部函数
conj	取共轭函数
isreal	是否为实数函数
round	四舍五入
fix	保留整数部分
floor	全舍
ceil	全入
sign	符号函数
rem	余数
mod	取模

2. 绘图函数

在 MATLAB 中绘图用 plot 函数完成，plot 函数的使用有以下四种形式：

- plot(y) 当 y 为向量时，以 y 的序号作为 X 轴，按向量 y 的值绘制曲线。
- plot(x,y)当 x,y 均为向量时，以 x 向量作为 X 轴，向量 y 作为 Y 轴绘制曲线。
- plot(x,y,s)该种命令形式不仅能做出类似于第二种形式的曲线，还可以对曲线的颜色、曲线上的点型，曲线线型进行定制。完成定制通过括号中的 s 完成。s 为表示颜色、点型及线型三种符号中的一种或多种组合。s 可采用的选择如表 1-6 所示。如：plot(x,y, 'r')表示做一条红色曲线，plot(x,y, 'c+')表示做一条青色曲线，曲线由一些离散的“+”点组成，plot(x,y, 'y+-')表示一条黄色的曲线，曲线上由 x 和 y 组合成的点由黄色“+”表示。
- plot(x1,y1,s1,x2,y2,s2,...) 同时定制几个曲线。

表 1-6 plot 函数 s 组成

颜 色	说 明	点 型	说 明	线 型	说 明
b	蓝色	.	点	-	实线
g	绿色	o	圆	:	点
r	红色	x	x 点	-.	点划线
c	青色	+	加号	--	划线



续表

颜 色	说 明	点 型	说 明	线 型	说 明
m	品红	*	星号	(none)	没有线型
y	黄色	s	方块		
k	黑色	d	宝石符号		
		v	下三角符号		
		^	上三角符号		
		<	左三角符号		
		>	右三角符号		
		p	五边形		
		h	六边形		

注意：在 MATLAB 绘图中默认是蓝色，多种颜色循环使用。

此外为对图形进行进一步操作，MATLAB 还提供了一些常用的图形操作函数，如表 1-7 所示。

表 1-7 MATLAB 常用图形操作函数

函 数	说 明
stem	离散曲线
bar	阶梯图
errorbar	误差条形图
hist	直方图
fill	在曲线和坐标轴之间的封闭区填以指定的颜色
polar	极坐标绘图
loglog	双对数 X-Y 坐标绘图
semilogx	半对数 X 坐标绘图
semilogy	半对数 Y 坐标绘图
plotyy	用左右两种 Y 坐标绘图
axis	控制坐标轴比例和外观
hold	保持当前图形
subplot	在平铺位置建立图形轴系
title	标出图形名称
xlabel	X 轴标注
ylabel	Y 轴标注
text	在图上标文字
gtext	用鼠标定位文字
legend	标注图例
grid	图上加坐标网格



1.1.4 文件建立

MATLAB 工作方式有两种：

- 交互式的指令操作方式。即用户在命令窗口中输入命令并按下回车键后，系统执行该指令并立即给出运算结果。
- M 文件的编程方式。M 文件是由 MATLAB 语句构成的文件，且文件名必须以.m 为扩展名。

其中交互式的指令操作方式编写灵活方便，易操作，但不利于规模集成，移植性和重用性也不好，因此一般在工程仿真实例中正常采用 M 文件的编程方式。这里对 M 文件编程中经常使用的程序流程控制和 M 函数编写予以介绍。

1. 程序流程控制

MATLAB 中程序流程控制有三种方式：

- 顺序结构：MATLAB 从上到下依次执行各语句，该结构最简单。
- 循环结构：循环结构有两种语法：for-end 和 while-end。
- 分支结构：分支结构用 if-end 完成。

注意：MATLAB 流程控制三种结构可嵌套使用。例如分支结构中可出现循环结构，循环结构中也可出现分支结构。

2. M 函数文件编写

M 函数文件与一般的 M 文件不同之处在于其表头存在一句命令行，其语法格式如下：

```
function [out1,out2,...] = funname(in1,in2,...)
```

其中 function 是固定语句，表示是 M 函数；

in1,in2,...是输入变量名，可根据需要制定；

out1,out2,...是输出变量名，可根据需要制定；

funname 是函数名，可根据需要制定。

注意：首先 M 函数文件名和函数名尽量存成一样，其次文件保存的路径一定要是 MATLAB 能够找到的路径，如果文件数量较少，建议放于当前工作路径下。

1.2 船舶运动动力学及控制器

将船舶视为刚体，其动力学在平衡点经 3 阶展开如式 (1-1) 和式 (1-2) 所示：

$$\begin{bmatrix} m - X_{\dot{u}} & 0 & 0 \\ 0 & m - X_{\dot{v}} & mx_g - Y_{\dot{r}} \\ 0 & mx_g - N_{\dot{v}} & I_z - N_{\dot{r}} \end{bmatrix} \begin{bmatrix} \Delta \dot{u} \\ \dot{v} \\ \dot{r} \end{bmatrix} = \begin{bmatrix} X \\ Y \\ N \end{bmatrix} \quad (1-1)$$



$$\begin{cases} X = X_u \Delta u + X_{uu} \Delta u^2 + X_{uuu} \Delta u^3 + X_{vv} \Delta v^2 + X_{rr} \Delta r^2 + X_{rv} r v \\ \quad + X_{\delta\delta} \Delta \delta^2 + X_{u\delta\delta} \Delta u \Delta \delta^2 + X_{v\delta} v \Delta \delta + X_{uv\delta} \Delta u v \Delta \delta \\ Y = Y_v v + Y_r r + Y_{vv} v^3 + Y_{vvr} v^2 r + Y_{vu} \Delta u + Y_{ru} r \Delta u \\ \quad + Y_{\delta} \Delta \delta + Y_{\delta\delta\delta} \Delta \delta^3 + Y_{u\delta} \Delta u \Delta \delta + Y_{uu\delta} \Delta u^2 \Delta \delta + Y_{v\delta\delta} v \Delta \delta^2 + Y_{vv\delta} v^2 \Delta \delta \\ N = N_v \Delta v + N_r \Delta r + N_{vvv} \Delta v^3 + N_{vu} v \Delta u + N_{ru} r \Delta u + N_{\delta} \Delta \delta \\ \quad + N_{\delta\delta\delta} \Delta \delta^3 + N_{u\delta} \Delta u \Delta \delta + N_{uu\delta} \Delta u^2 \Delta \delta + N_{v\delta\delta} v \Delta \delta^2 + N_{vv\delta} v^2 \Delta \delta \end{cases} \quad (1-2)$$

式中 $[\Delta u \quad v \quad r]$ 分别为纵向速度偏差、横漂速度偏差和舵向速度偏差, $\Delta u = u - U_0$, $\Delta v = v$, $\Delta r = r$, U_0 为经济航速, m 为质量, I_z 为转动惯量, x_g 为中心重心距, $[X \quad Y \quad N]$ 为舵力, X_u , X_{uu} , Y_v , N_v 等是动力学泰勒展开时的系数。

船舶运动学方程如式(1-3)所示:

$$\begin{cases} \dot{x} = u \cos \psi - v \sin \psi + U_c \cos \gamma_c \\ \dot{y} = u \sin \psi + v \cos \psi + U_c \sin \gamma_c \\ \dot{\psi} = r \\ \dot{\delta} = \dot{\delta} \end{cases} \quad (1-3)$$

式中 ψ 为偏航角, U_c 为流速, γ_c 为流向, x 为北向坐标, y 为东向坐标, δ 为舵偏角。

控制器方程如式(1-4)所示:

$$\delta = \begin{cases} |\delta| < \delta_{\max} & -K_p ((\psi - \psi_0) + T_d \times r) \\ |\delta| > \delta_{\max} & \delta_{\max}, -\delta_{\max} \end{cases} \quad (1-4)$$

式中 δ 为舵偏角, δ_{\max} 为允许最大舵偏角, K_p 为 PD 控制器 P 增益, T_d 为 PD 控制器 D 增益, ψ_0 为目标偏航角。

船舶控制逻辑为: 通过控制舵偏角使得船舶偏航角达到指定角度, 控制器采用 PD 算法。

1.3 船舶运动控制器设计及仿真程序

在上一节介绍船舶运动动力学基础上, 本节给出基于 MATLAB 的仿真程序。

程序由两个 M 文件组成: boat_PD.m 和 mariner.m。其中 boat_PD.m 为主程序, 完成状态变量及其他变量的赋初值和在一定仿真时间内利用欧拉迭代法对基于 PD 控制的船舶运动进行仿真, 以及通过作图对仿真结果进行显示。而 mariner.m 为 M 函数文件, 完成船舶运动动力学、运动学及控制器建模, 其作用是供 boat_PD.m 调用, 完成仿真。具体代码如下所示:

```
% 主函数, 文件名为 boat_PD
t_f = 600;      % 仿真事件设定
h   = 0.1;      % 采样时间
```




```

Kp = 1;      % 控制器 P 增益
Td = 10;     % 控制器 D 增益

% 状态 x = [ u v r x y psi delta ]' 赋初值
x = zeros(7,1);

N = round(t_f/h);      % 采样量
xout = zeros(N+1,length(x)+2); % 输出变量赋初值

% 分支结构流程控制
for i=1:N+1,
    time = (i-1)*h;
    r = x(3);
    psi = x(6);

    psi_ref = 5*(pi/180); % 控制目标角度
    delta = -Kp*(psi-psi_ref)+Td*r; % PD 控制器

    % 调用 M 函数文件
    [xdot,U] = mariner(x,delta); % 船舶模型

    % 存储数据以便后续调用
    xout(i,:) = [time,x',U];

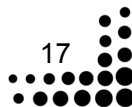
    % 数值积分, 欧拉算法
    x = x + h*xdot
end

% 从存储的数据中给变量赋值
t = xout(:,1);
u = xout(:,2);
v = xout(:,3);
r = xout(:,4)*180/pi; % pi 为 MATLAB 特殊常量, 表示圆周率
x = xout(:,5);
y = xout(:,6);
psi = xout(:,7)*180/pi;
delta = xout(:,8)*180/pi;
U = xout(:,9);

% 作图
% 如果要作多个图, 用 figure(i), i = 1, 2, 3, ... 来实现
figure(1)
% 作完图之后, 利用 axis, xlabel 等来丰富和定制图形的信息
plot(y,x),grid,axis('equal'),xlabel('East'),ylabel('North'),title('Ship position')

figure(2)
% 如果要求在一个图中作多个小图, 用 subplot 来完成
subplot(221),plot(t,r),xlabel('time (s)'),title('yaw rate r (deg/s)'),grid

```





```
subplot(222),plot(t,U),xlabel('time (s)'),title('speed U (m/s)'),grid
subplot(223),plot(t,psi),xlabel('time (s)'),title('yaw angle \psi (deg)'),
grid
subplot(224),plot(t,delta),xlabel('time (s)'),title('rudder angle \delta
(deg)'),grid
```

```
% M函数文件，文件名为 mariner
function [xdot,U] = mariner(x,ui,U0)
% [xdot, U] = mariner(x,ui) 返回真实速度 U in m/s 以及状态变量 x = [ u v r x
y psi delta n ]'
%的偏差
%
%输入变量有：
% u    = 与常规速度  $U_0$  之间的偏差， $U_0$  默认值为  $U_0 = 7.7175 \text{ m/s} = 15 \text{ knots}$ 。
% v    = 摇摆速度与 0 之间的偏差 (m/s)
% r    = 偏航角速度与 0 之间的偏差 (rad/s)
% x    = x 方向的位置 (m)
% y    = y 方向的位置 (m)
% psi  = 偏航角与 0 之间的偏差 (rad)
% delta = 真实的舵偏角 (rad)
% ui    = 控制舵角指令(rad)，即控制器输出
% U0    = 常规速度 (m/s)

% 输出变量
% xdot 为状态变量的微分
% U    = 真实速度 (m/s)

% 确认输入值是否符合要求
% 如果输入不是 7 维，则输出错误信息
if (length(x) ~= 7),error('x-vector must have dimension 7 !'); end
% 如果控制舵角指令不是 1 维，则输出错误信息
if (length(ui) ~= 1),error('ui must be a scalar input!'); end
% 如果没有输入  $U_0$  信息，则采用默认值
if nargin==2, U0 = 7.7175; end

% 对变量赋值及归一化
L = 160.93;
U = sqrt((U0 + x(1))^2 + x(2)^2); % 真实速度
delta_c = -ui;
u = x(1)/U;
v = x(2)/U;
r = x(3)*L/U;
psi = x(6);
delta = x(7);
delta_max = 40; % 最大舵角 (deg)
Ddelta_max = 5; % 最大舵角速度 (deg/s)
m = 798e-5;
Iz = 39.2e-5;
xG = -0.023;
```



% 舵力及力矩在 0 值附近展开系数赋值

```
Xudot = -42e-5; Yvdot = -748e-5; Nvdot = 4.646e-5;
Xu = -184e-5; Yrdot = -9.354e-5; Nrdot = -43.8e-5;
Xuu = -110e-5; Yv = -1160e-5; Nv = -264e-5;
Xuuu = -215e-5; Yr = -499e-5; Nr = -166e-5;
Xvv = -899e-5; Yvvv = -8078e-5; Nvvv = 1636e-5;
Xrr = 18e-5; Yvvr = 15356e-5; Nvvr = -5483e-5;
Xdd = -95e-5; Yvu = -1160e-5; Nvu = -264e-5;
Xudd = -190e-5; Yru = -499e-5; Nru = -166e-5;
Xrv = 798e-5; Yd = 278e-5; Nd = -139e-5;
Xvd = 93e-5; Yddd = -90e-5; Nddd = 45e-5;
Xuvd = 93e-5; Yud = 556e-5; Nud = -278e-5;
        Yuud = 278e-5; Nuud = -139e-5;
        Yvdd = -4e-5; Nvdd = 13e-5;
        Yvvd = 1190e-5; Nvvd = -489e-5;
        Y0 = -4e-5; N0 = 3e-5;
        Y0u = -8e-5; N0u = 6e-5;
        Y0uu = -4e-5; N0uu = 3e-5;
```

% 等式 (1-1) 左边变量

```
m11 = m-Xudot;
m22 = m-Yvdot;
m23 = m*xG-Yrdot;
m32 = m*xG-Nvdot;
m33 = Iz-Nrdot;
```

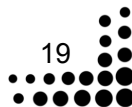
% M 文件流程控制中的分支结构

```
if abs(delta_c) >= delta_max*pi/180,
% 如果计算的舵角大于舵角允许最大值, 则采用舵角允许最大值
    delta_c = sign(delta_c)*delta_max*pi/180;
end
% 舵角速度等于现在的舵角值减去上次舵角值
delta_dot = delta_c - delta;
if abs(delta_dot) >= Ddelta_max*pi/180,
% 如果计算的舵角速度大于舵角速度允许最大值, 则采用舵角速度允许最大值
    delta_dot = sign(delta_dot)*Ddelta_max*pi/180;
end
```

% 等式 (1-2)

```
X = Xu*u + Xuu*u^2 + Xuuu*u^3 + Xvv*v^2 + Xrr*r^2 + Xrv*r*v + Xdd*delta^2
+ ...
    Xudd*u*delta^2 + Xvd*v*delta + Xuvd*u*v*delta;
Y = Yv*v + Yr*r + Yvvv*v^3 + Yvvr*v^2*r + Yvu*v*u + Yru*r*u + Yd*delta
+ ...
    Yddd*delta^3 + Yud*u*delta + Yuud*u^2*delta + Yvdd*v*delta^2 + ...
    Yvvd*v^2*delta + (Y0 + Y0u*u + Y0uu*u^2);
N = Nv*v + Nr*r + Nvvv*v^3 + Nvvr*v^2*r + Nvu*v*u + Nru*r*u + Nd*delta
+ ...
    Nddd*delta^3 + Nud*u*delta + Nuud*u^2*delta + Nvdd*v*delta^2 + ...
    Nvvd*v^2*delta + (N0 + N0u*u + N0uu*u^2);

detM22 = m22*m33-m23*m32;
```





% 分量计算

```
xdot = [ X*(U^2/L)/m11  
          - (-m33*Y+m23*N)*(U^2/L)/detM22  
          (-m32*Y+m22*N)*(U^2/L^2)/detM22  
          (cos(psi)*(U0/U+u)-sin(psi)*v)*U  
          (sin(psi)*(U0/U+u)+cos(psi)*v)*U  
          r*(U/L)  
          delta_dot ];
```

仿真结果如图 1-1 和图 1-2 所示。其中图 1-1 为动力学仿真结果，图中左上角曲线表示偏航角速度，左下角曲线表示偏航角，右上角为船舶真实速度曲线，右下角曲线为舵偏角。而图 1-2 为运动学仿真结果，图中曲线显示了船舶在实际坐标系中的运动轨迹。

由图可看出，文中设计的控制器能够使船舶偏航角达到指定角，控制器是有效的。

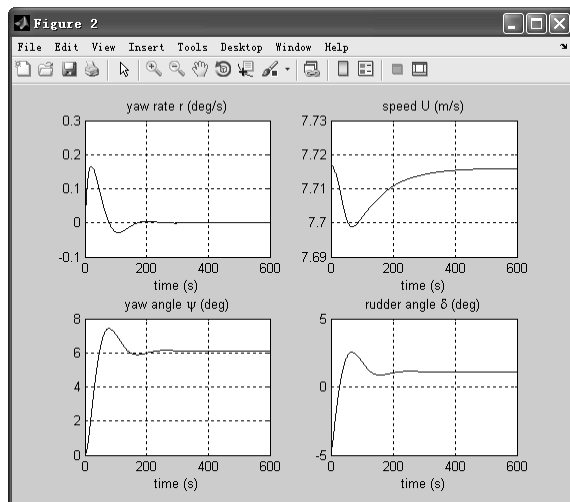


图 1-1 船舶运动动力学仿真结果

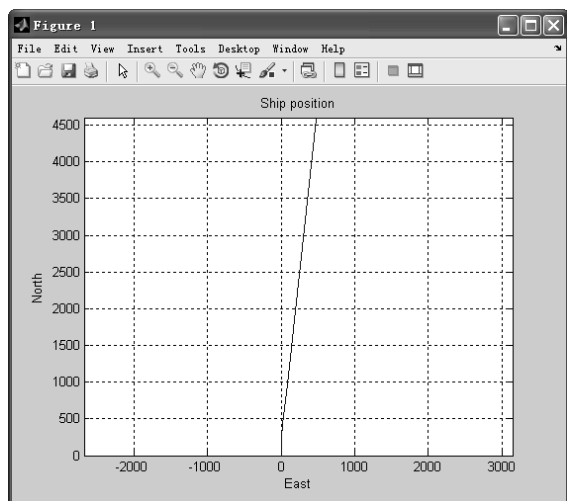


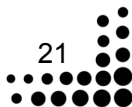
图 1-2 船舶运动运动学仿真结果

1.4 本例小结

本例首先介绍了变量、运算符、数学函数和文件建立等 MATLAB 编程基础。然后通过船舶运动控制实例来展示 MATLAB 编程的技巧和基于 PD 算法设计控制器的方法。

通过本例的学习，读者应该掌握以下几个知识点：

- 了解变量、运算符、数学函数等概念。
- 掌握基于 M 文件的编程方法，对流程控制和 M 函数文件编写应能熟练运用。
- 掌握 MATLAB 作图函数及相关的操作函数。
- 掌握基于 PD 算法设计控制器的方法。





第 2 例 F-14 战斗机俯仰轴控制仿真

F-14 是根据美国海军 70 年代至 80 年代舰队防空和护航的要求，由格鲁曼公司研制的双座超音速多用途舰载战斗机。本例基于 MATLAB 和采用 PI 算法对 F-14 战斗机俯仰轴运动设计控制器，并进行仿真。在设计之前，为使读者熟悉基于 Simulink 的设计与仿真过程，首先对其进行简单介绍，其次介绍 F-14 战斗机俯仰轴运动动力学，并进行控制器设计，最后给出仿真程序。

通过本例学习应了解和掌握以下几部分：

- Simulink 建模和仿真基础。
- F-14 俯仰轴动力学与控制器设计。

2.1 Simulink 建模及仿真基础

Simulink 与基于 MATLAB 语言编程类似，Simulink 的基本要素也可大致分为变量、运算符、函数及文件四类。Simulink 中变量、运算符、函数都以模块的形式给出，这三类对象通过组合实现对工程对象的建模；Simulink 文件是以 .mdl 结尾的文件类型。MATLAB 语言编程与 Simulink 仿真在大部分情况下是可以相互转换的。

基于 MATLAB 语言和基于 Simulink 建模仿真的区别：

- 基于 Simulink 建模仿真必须定义仿真时间，即 Simulink 是按照时间轴利用解算器往前推进的，而基于 MATLAB 语言编程不仅能完成仿真功能，对于没有时间轴特性的对象及事件也可以建模。
- 部分 MATLAB 语言支持的数据类型在 Simulink 不能使用，如元胞数组等。

可以说，Simulink 能实现的功能在 MATLAB 语言中都能实现，而在 MATLAB 语言中能实现的部分功能在 Simulink 中却不能实现。尽管如此，Simulink 相对于 MATLAB 语言有更加直观、操作性强、易于理解和建模简单方便等优点，在实际工程仿真中很多采用基于 Simulink 的方式。

下面对基于 Simulink 的建模仿真流程予以介绍。

Simulink 是一个对动态系统进行建模、仿真和分析的软件包。它支持线性和非线性系统在连续、离散或两种混合模式下仿真，此外系统还可以是多采样率。在建模上，Simulink 提供了一个图形化用户界面，可以拖拽模块图标进行建模。从建模角度讲，它既适于自上而下设计流程，又适于自下而上的逆程设计。

为使用 Simulink，首先需要打开它，打开 Simulink 有两种方式：一种是在 MATLAB 命令行中输入“Simulink”，另一种是在 MATLAB 工具栏中 Simulink 按钮打开。打开后

运行界面如图 2-1 所示。由图可看出，通过单击新建模型按钮即可打开一个空白 Simulink 文件，即后缀为.mdl 的文件，在上面拖拽模块并连接即可完成建模工作。下面对常用的 Simulink 模块予以介绍。

1. 输出模块

输出模块如图 2-2 所示。图中常用的有示波器和输出端口，示波器能够显示变量随时间变化曲线，而输出端口在子模块建立中起输出作用。

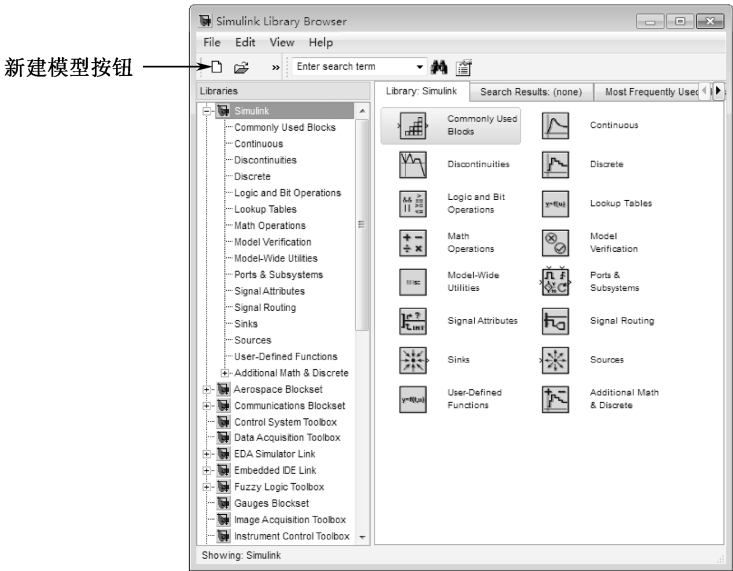


图 2-1 Simulink 运行界面

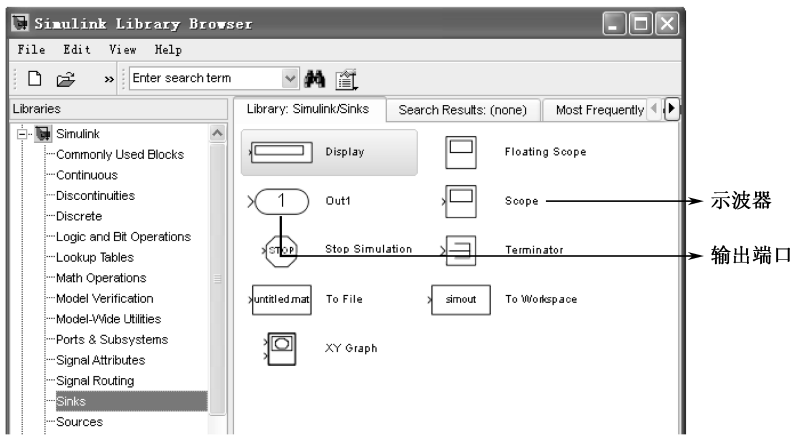


图 2-2 输出模块示意图

2. 连续系统模块

连续系统模块如图 2-3 所示，利用图中模块可完成对连续系统的建模，其中常用的有



积分模块、状态方程模块和传递函数模块等。其中积分模块在微分方程建立中经常用到，状态方程模块和传递函数模块在线性系统建模时也经常用到。

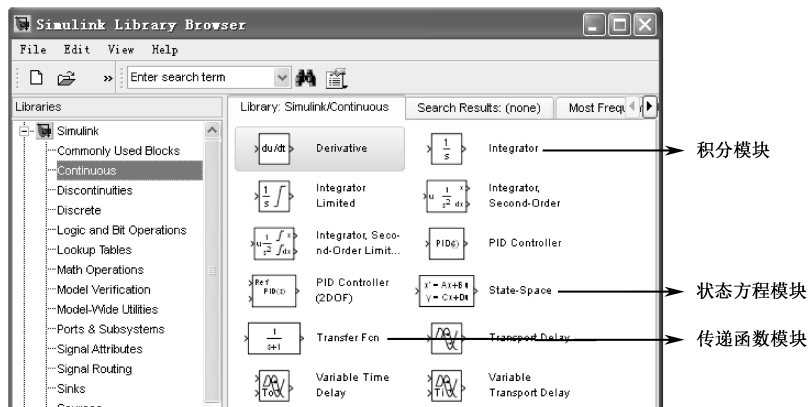


图 2-3 连续系统模块

3. 数据源模块

数据源模块如图 2-4 所示，数据源模块为系统建模提供各种数据，如常量、随机数和阶跃信号等，其中常用的有白噪声模块、常值模块、输入模块、信号发生器模块和阶跃信号模块。其中输入模块和前面的输出模块经常成对使用，放置于子系统模块中，作为子系统与外部模块连接的端口。



图 2-4 数据源模块

4. 系统及端口模块

系统及端口模块如图 2-5 所示，其中经常用到子系统模块。子系统模块的功能类似于 M 函数文件，通过子系统封装能够使系统简洁明了，更具可读性。

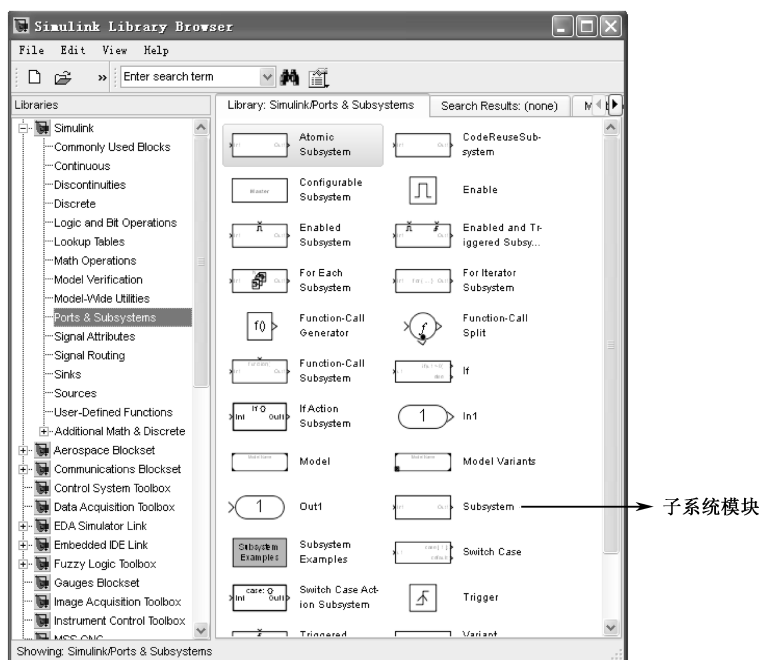


图 2-5 系统及端口模块

说明：在 Simulink 中还有很多模块，由于篇幅限制不予详细列出，感兴趣的读者可自行分析研究，此外读者还可以针对特定领域建立自己的 Simulink 模块库，这样在设计时将能提高设计效率，如何设计用户自己的 Simulink 模块库在后面的章节将予以介绍。

2.2 F-14 俯仰轴动力学模型

F-14 俯仰轴动力学由等式 (2-1) 和式 (2-2) 组成。其中式 (2-1) 表示 F-14 在垂直方向动力学，式 (2-2) 表示 F-14 在俯仰轴方向动力学。

$$\dot{w} - Z_w w = Z_d d + U_0 q - W_{gust} \quad (2-1)$$

$$\dot{q} - M_q q = M_w w + M_d d - q_{gust} \quad (2-2)$$

式中 w 表示垂直方向速度， d 表示飞机升降舵实际偏差角， q 表示俯仰轴角速度， W_{gust} 表示在垂直方向的风载， q_{gust} 表示在俯仰轴方向的风载， Z_w 、 Z_d 、 U_0 、 M_q 、 M_w 和 M_d 为相应的系数。

飞机实际升降舵偏差角 d 是飞机根据控制器指令 c 得到，两者之间关系如式 (2-3) 所示：



$$T_a \dot{d} + d = c \quad (2-3)$$

式中 T_a 为执行机构惯性系数，表征飞机升降舵延迟特性。

控制指令 c 是将偏差 e 作为输入，按照 PI 控制算法计算得到的控制数值，如式 (2-4) 所示：

$$c = K_f e + K_i \int e \quad (2-4)$$

式中 K_f 表示 PI 控制算法中 P 增益， K_i 表示 PI 控制算法中 I 增益。

e 由飞行员指令 u ，攻角 α 和俯仰角 q 组成，它们之间关系如式 (2-5) 所示：

$$e = \frac{1}{T_s s + 1} u - \frac{K_a}{T_{al} s + 1} \alpha - K_q \frac{s + w_1}{s + w_2} q \quad (2-5)$$

式中 $\frac{1}{T_s s + 1}$ 、 $\frac{K_a}{T_{al} s + 1}$ 和 $K_q \frac{s + w_1}{s + w_2}$ 为相应变量的滤波器。

风载 W_{gust} 和 q_{gust} 可由式 (2-6) 求得：

$$\begin{cases} W_g = \frac{S_{wg}}{\sqrt{a^3}} \frac{\sqrt{3}as + 1}{s^2 + \frac{2}{a}s + \frac{1}{a^2}} f_{noise} \\ Q_g = \frac{pi}{4b} \frac{s}{s + \frac{piV_{t0}}{4b}} W_g \\ W_{gust} = Z_w W_g \\ q_{gust} = M_w W_g + M_q Q_g \end{cases} \quad (2-6)$$

式中 f_{noise} 为白噪声。

由式 (2-6) 可见，俯仰轴风载与垂直方向风载耦合。

此外，攻角 α 和垂直方向速度 w 关系如式 (2-7) 所示。

$$\alpha = \frac{1}{U_0} w \quad (2-7)$$

2.3 基于 Simulink 的 F-14 俯仰轴仿真模型

基于 Simulink 的 F-14 俯仰轴仿真模型如图 2-6 至图 2-9 所示。其中图 2-6 给出整体框图，图 2-7 给出 F-14 在垂直方向和俯仰轴方向动力学模型框图，图 2-8 给出控制器模型框图，图 2-9 给出风载模型框图。

1. 整体框图

整体框图由三个子系统以及一些其他模块组成。三个子系统为：Aircraft Dynamics Model (F-14 在垂直方向和俯仰轴方向动力学模型)、Controller (控制器模型) 和 Dryden Wind Gust Models (风载)。子系统模块通过在图 2-5 中子系统模块内部添加适当的模块来实现。

2. F-14 在垂直方向和俯仰轴方向动力学模型框图

The screenshot shows the Simulink model 'demo_f14'. The model is a control system for an F-14 aircraft. The main components and signal flow are as follows:

- Pilot:** A block representing the pilot's input, which outputs a signal u to a summing junction.
- Stick Input:** A block that receives the pilot's input u and outputs a signal to the Controller.
- Controller:** A block that receives the Stick Input and outputs a signal to the Actuator Model.
- Actuator Model:** A block that receives the signal from the Controller and outputs a signal to the Aircraft Dynamics Model.
- Aircraft Dynamics Model:** A block that receives the signal from the Actuator Model and outputs several signals:
 - δ (deg): Elevator Deflection
 - w (ft/sec): Vertical Velocity
 - q (rad/sec): Pitch Rate
 - α (rad): Angle of Attack
- Dryden Wind Gust Models:** A block that receives the signal w and outputs signals w_{gust} and q_{gust} to the summing junction.
- Summing Junction:** A block that adds the signals from the Actuator Model and the Dryden Wind Gust Models to produce the final signal q .
- 1/s:** A block that integrates the signal q to produce the final output α (rad).
- Scopes:** Several scopes are used to monitor the system's performance:
 - Pilot G force (g):** A scope that monitors the pilot's g-force.
 - Nz pilot calculation:** A block that calculates the normal acceleration N_z of the pilot.
 - Pilot G force Scope:** A scope that monitors the pilot's g-force.
 - Nz Pilot (g):** A scope that monitors the normal acceleration N_z of the pilot.
 - Angle of Attack:** A scope that monitors the angle of attack α .

图 2-6 F-14 俯仰轴动力学模型

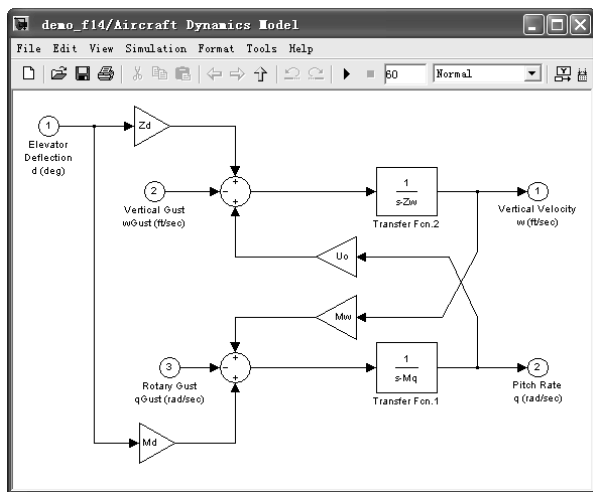
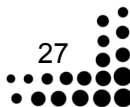


图 2-7 F-14 在垂直方向和俯仰轴方向动力学模型

3. 控制器模型框图

控制器框图中有三个输入和一个输出。三个输入分别为飞行员控制输入(Stick Input)，





攻角 (α) 和俯仰角速度 (q)，输入同样是由输入端口模块完成。三个输入的滤波器 (Stick Prefilter, Alpha-sensor Low-pass Filter, Pitch Rate Lead Filter) 是由传递函数模块实现，实现之后通过两个加减法模块实现三者之间的融合。

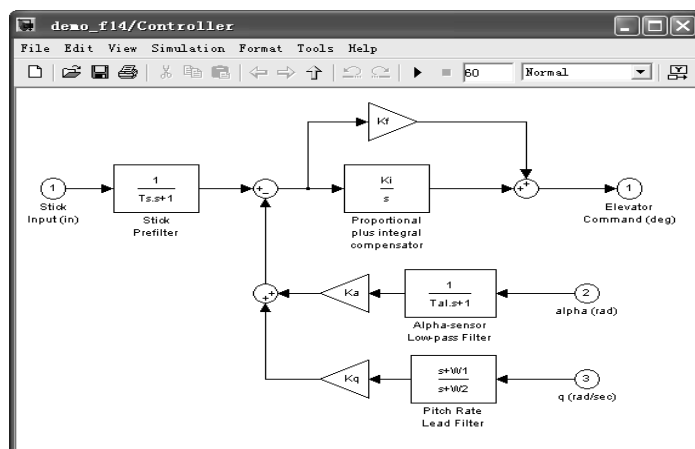


图 2-8 F-14 俯仰轴控制器模型

控制器框图的控制器 Proportional plus integral compensator 由两部分相加完成，其中 P 增益模块由比例模块实现，比例模块也可在“Simulink|Math Operations”中找到，而 I 增益模块由积分模块实现。

控制器框图的输出为 Elevator Command，由输出端口实现。

4. 风载模型框图

风载模型框图中信号源由白噪声模块实现，之后的计算由传递函数模块实现。

在完成 F-14 俯仰轴模型搭建后，如图 2-9 所示在仿真时间设定位置设置时间为 60 秒，然后单击仿真开始按钮，得到仿真结果如图 2-10 所示。图中左侧 (a) 表示飞行员输入指令曲线，右侧 (b) 表示输出攻角曲线。两者对比可见输出攻角能够实现对飞行员输入指令的跟踪，文中设计的 PI 控制器能够实现控制功能。

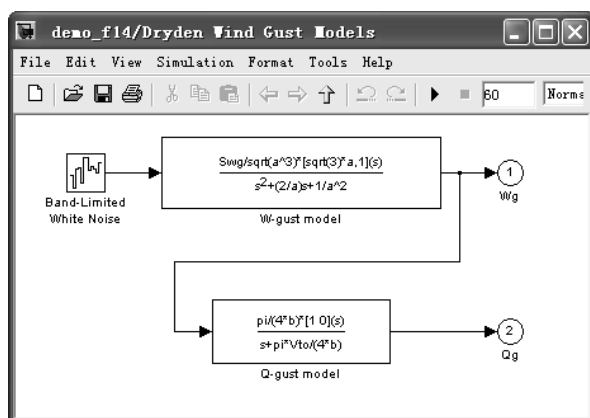
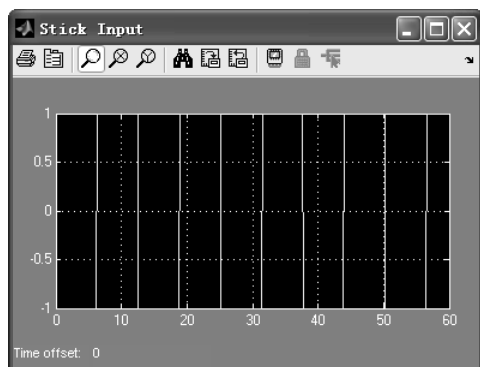
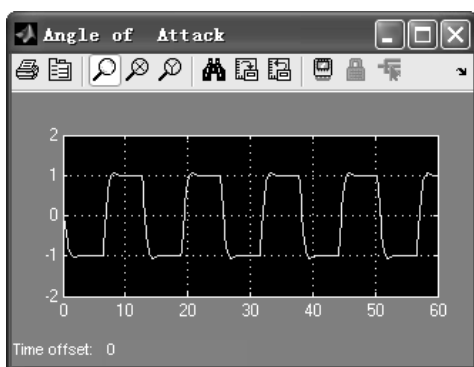


图 2-9 F-14 风载模型



(a) 输入指令



(b) 输出攻角

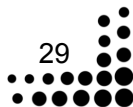
图 2-10 F-14 俯仰轴输入指令及输出攻角示意图

2.4 本例小结

本例首先介绍了基于 Simulink 建模仿真基础知识。然后通过 F-14 战斗机俯仰轴控制实例来展示 Simulink 建模技巧以及基于 PI 算法设计控制器的方法。

通过本例的学习，读者应该掌握以下几个知识点：

- 了解 Simulink 建模仿真过程。
- 熟练掌握 Simulink 常用模块使用方法，能够使用常用模块对动力学数学模型进行建模仿真。
- 掌握基于 PI 算法设计控制器过程。
- 了解 F-14 战斗机俯仰轴动力学特性。





第 3 例 汽车主动悬架控制器设计与仿真

汽车悬挂系统是汽车的车架与车桥或车轮之间的一切传力连接装置的总称，其作用是传递作用在车轮和车架之间的力和力矩，并且缓冲由不平路面传给车架或车身的冲击力，并衰减由此引起的振动，以保证汽车能平顺地行驶。典型的悬挂系统结构由弹性元件和减震器等组成。弹性元件又有钢板弹簧、空气弹簧、螺旋弹簧以及扭杆弹簧等形式。悬挂系统是汽车中的一个重要组成，它把车架与车轮弹性地联系起来，关系到汽车的多种使用性能。主动悬挂系统是近十几年发展起来的，其主要的特性是能够根据实际情况主动调整悬挂系统的负载，从而使得乘驾者获得最好的舒适度。主动悬挂系统一方面通过传感器测得车轮制动压力、转向盘角度以及转向速度等信息，另一方面通过执行元件使悬架系统处于合适的系统状态，从而满足驾驶性和舒适性的要求。

本例采用鲁棒算法对汽车悬架系统设计控制器，并进行仿真。在设计之前，为使读者了解汽车悬架系统的动力学特性以及与主动悬架系统对比，首先对被动悬架系统进行仿真，其次考虑没有不确定性条件下设计汽车主动悬架系统控制器，最后再考虑不确定性条件下设计汽车主动悬架系统控制器。

通过本例学习应了解和掌握以下几部分：

- 汽车被动悬架系统动力学。
- 汽车主动悬架系统动力学。
- 汽车主动悬架控制器设计。

3.1 汽车被动悬架系统仿真

这里汽车被动悬架系统采用一个简化的半车模型。模型包括前轮和后轮悬架系统以及车体在垂直方面上的倾斜以及径向距离自由度动力学。通过该模型分析被动悬挂系统及其对车体影响的动力学特性。

如图 3-1 所示半车模型示意图。其中汽车前后悬架系统可建模为弹簧-阻尼系统。如果想要建立更详细的模型，可以考虑加入轮胎模型和非线性阻尼特性（与速度相关的阻尼力）。系统具有两个自由度：倾斜角度和与地面距离。系统共有四个状态变量：垂直距离、垂直速度、倾斜角度和倾斜角速度。

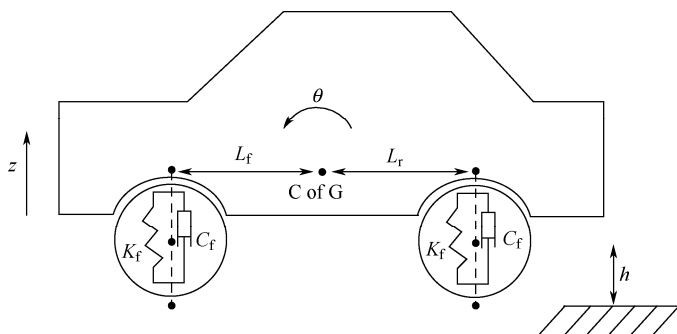


图 3-1 半车模型示意图

3.1.1 被动悬架系统动力学模型

被动悬架系统由等式 (3-1) 至式 (3-5) 组成：

$$F_{front} = 2K_f(L_f\theta - z) + 2C_f(L_f\dot{\theta} - \dot{z}) \quad (3-1)$$

式中 F_{front} 为由前悬架系统作用与汽车上的力；

K_f 为前悬架系统弹性系数；

C_f 为前悬架系统阻尼系数；

L_f 为前悬架系统与汽车重心间距离；

$\theta, \dot{\theta}$ 为倾斜角及角速度；

z, \dot{z} 为距地面垂直距离及速度。

$$M_{front} = -L_{front}F_{front} \quad (3-2)$$

式 (3-2) 描述了前轮悬架系统作用于汽车上的力矩。

$$F_{rear} = -2K_r(L_r\theta + z) - 2C_r(L_r\dot{\theta} + \dot{z}) \quad (3-3)$$

式中 F_{rear} 为由后悬架系统作用于汽车上的力；

K_r 为后悬架系统弹性系数；

C_r 为后悬架系统阻尼系数；

L_r 为后悬架系统与汽车重心间距离。

$$M_{rear} = L_rF_{rear} \quad (3-4)$$

式 (3-4) 描述了后轮悬架系统作用于汽车上的力矩。

$$\begin{cases} m_b\ddot{z} = F_{front} + F_{rear} - m_bg \\ I_{yy}\ddot{\theta} = M_{front} + M_{rear} + M_y \end{cases} \quad (3-5)$$

式中 m_b 为汽车质量；

M_y 是由垂直加速度引起的倾斜动量矩；

I_{yy} 是汽车转动惯量。



3.1.2 被动悬架系统 Simulink 模型

汽车被动悬架系统 Simulink 模型如图 3-2 和图 3-3 所示。其中图 3-2 为汽车被动悬架系统模型框图，图 3-3 为两自由度弹簧-阻尼模型框图。

汽车被动悬架系统模型框图中存在两个子系统，子系统的结构相同，都是为两自由度弹簧-阻尼模型。除子系统外，系统模型使用两个阶跃模块（Pitch moment induced by vehicle acceleration 和 Road Height）来作为输入，模拟汽车加速度干扰和地面障碍干扰，再通过两组双积分环节得到垂直高度（ Z ）和俯仰角（ Θ ），将它们反馈到子系统模块，即完成系统闭环。

两自由度弹簧-阻尼模型框图对应等式（3-1）和等式（3-2），分别得到前后悬架系统作用于汽车上的力。

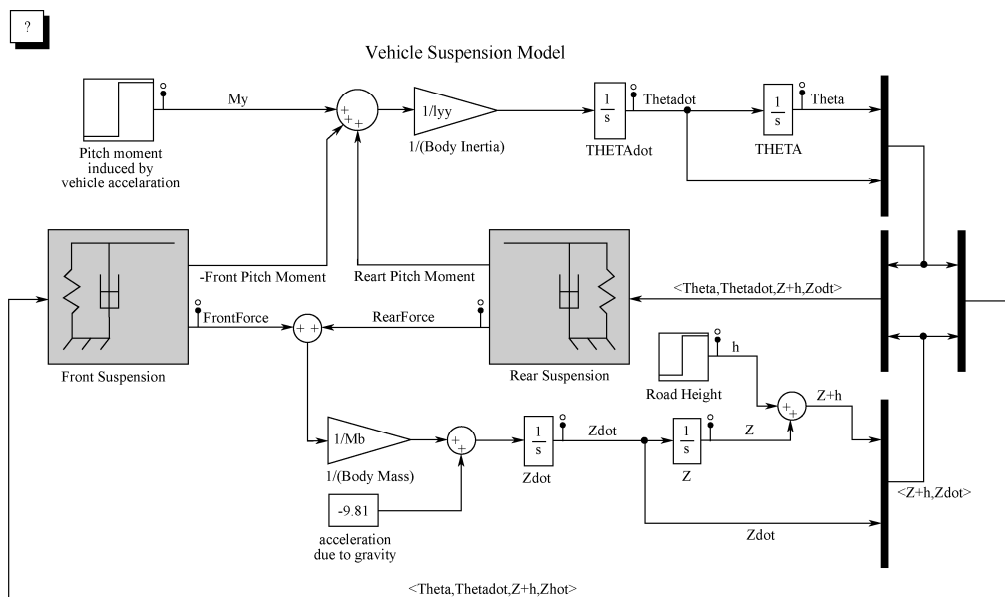


图 3-2 汽车被动悬架系统模型框图

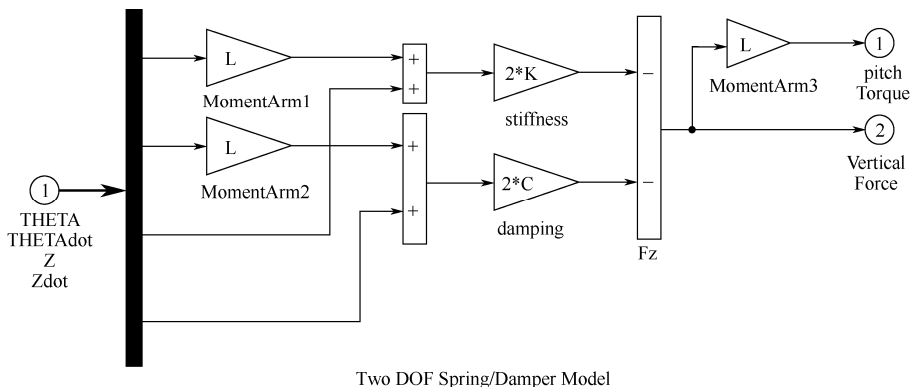


图 3-3 两自由度弹簧-阻尼模型框图

完成建模后，设置仿真时间为 10 秒，并通过在 MATLAB 命令行中输入以下语句对模型中相关变量进行赋值：

```
Lf = 0.9;  
Lr = 1.2;  
Mb = 1200;  
Iyy = 2100;  
kf = 28000;  
kr = 21000;  
cf = 2500;  
cr = 2000.
```

单击仿真开始按钮，得到仿真结果，如图 3-4 所示。由图可见，汽车被动悬架系统能够对外界干扰起到一定的削弱和抵制作用，为乘客提供一定舒适的乘驾环境。

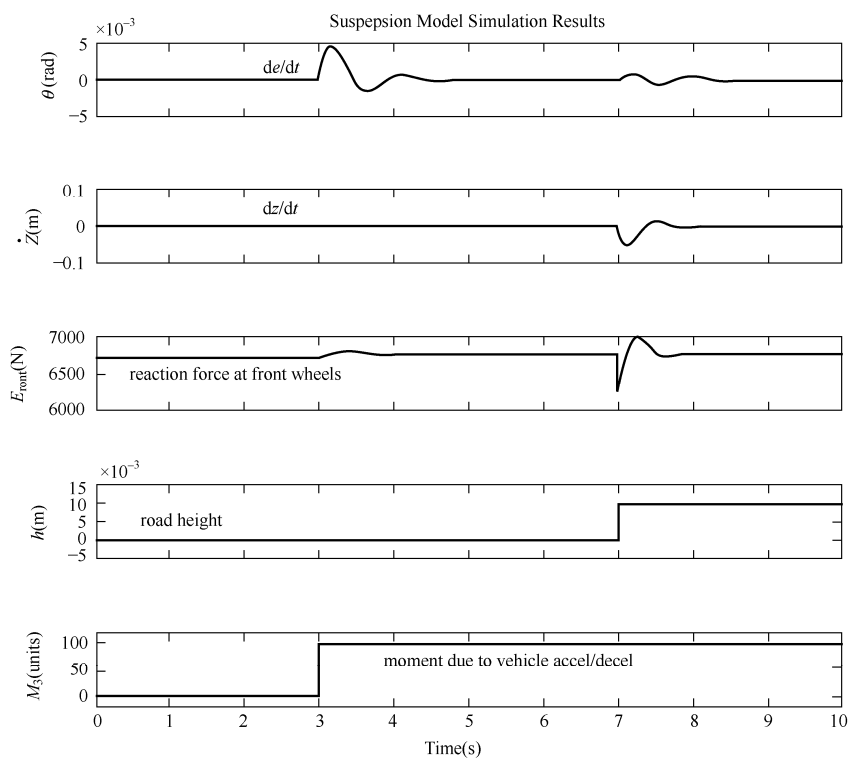


图 3-4 汽车被动悬架系统仿真曲线

3.2 汽车主动悬架系统控制器设计

3.2.1 主动悬架系统动力学模型

这里汽车主动悬架系统采用一个简化的 1/4 车模型，如图 3-5 所示。由图可见主动悬

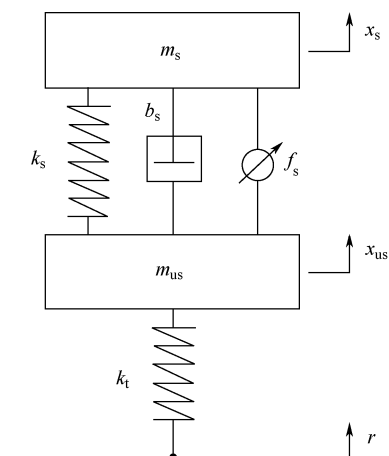


图 3-5 汽车主动悬架系统示意图

架系统与被动悬架系统不同之处在于：

- 主动悬架中存在主动控制力 f_s ，通过设计控制器，使 f_s 按照特定的规律作用力，能够更为有效地削弱外部干扰对汽车动力学影响，从而能够为乘客或驾驶员提供更为舒适的乘驾条件。
- 主动悬架系统分为两层：一层为单自由度弹簧模型，另一层为两自由度弹簧-阻尼模型，而被动悬架系统只有一层。相比而言，主动悬架系统更能够减小外部扰动对汽车的影响。

图中 m_s 表示汽车及乘客质量， m_{us} 表示车轮质量， k_s 和 k_t 为弹簧弹性系数， b_s 表示阻尼器阻尼系数， x_s, x_{us}, r 分别表示车体运动、车轮运动和路面干扰。

对图 3-5 给出的主动悬架系统进行建模得动力学模型如式 (3-6) 所示。

$$\begin{cases} \dot{x}_1 = x_2 \\ \dot{x}_2 = -\frac{1}{m_s} [k_s(x_1 - x_3) + b_s(x_2 - x_4) - f_s] \\ \dot{x}_3 = x_4 \\ \dot{x}_4 = \frac{1}{m_{us}} [k_s(x_1 - x_3) + b_s(x_2 - x_4) - k_t(x_3 - r) - f_s] \end{cases} \quad (3-6)$$

式中 $x_1 = x_s$ ， $x_2 = \dot{x}_s$ ， $x_3 = x_{us}$ ， $x_4 = \dot{x}_{us}$ 。

3.2.2 主动悬架系统控制器设计及建模仿真

为采用鲁棒控制算法设计主动悬架系统控制器，首先建立主动悬架系统增广模型，如图 3-6 所示。

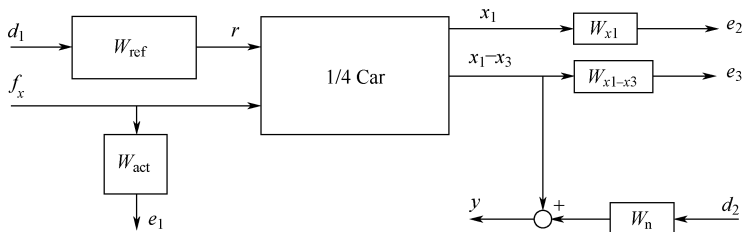


图 3-6 主动悬架系统增广模型

图中 d_1 表示地面干扰， W_{ref} 用于模拟地面干扰的量级， d_2 表示观测噪声， W_n 用于设定观测噪声值， W_{act} 对执行机构进行建模， W_{x1} 用于设定汽车在垂直方向控制精度， W_{x1-x3} 用于设定悬架系统控制精度。

建立增广模型之后即可以编写程序对其进行求解。程序代码如下：



```

% 首先对相关变量进行赋值
ms = 290;
mus = 59;
bs = 1000;
ks = 16182;
kt = 190000;

% 建立状态方程
A12 = [ 0 1 0 0; [-ks -bs ks bs]/ms] ;
A34 = [ 0 0 0 1; [ks bs -ks -kt -bs]/mus];
B12 = [ 0 0; 0 1000/ms];
B34 = [ 0 0; [kt -10000]/mus];
C = [1 0 0 0; A12(2,:); 1 0 -1 0; 0 0 0 0];
D = [0 0; B12(2,:); 0 0; 0 1];
qcar = ss([A12; A34], [B12; B34], C, D) % 使用 ss 函数, 具体用法请见 A1

% 建立增广模型
% 首先为建立增广模型赋值
Wn = 0.01;
Wref = 0.07;
Wact = (100/13)*tf([1 50], [1 500]);
Wx1 = 8*tf(2*pi*5, [1 2*pi*5]);

Systemnames = 'qcar Wn Wref Wact Wx1'; % 状态变量
Inputvar = '[d1; d2; fs]'; % 输入变量
Outputvar = 'Wact; Wx1; qcar(3)+Wn'; % 输出变量

% 建立系统及变量间关系
input_to_qcar = '[Wref; fs]';
input_to_Wn = '[ d2 ]';
input_to_Wref = '[ d1 ]';
input_to_Wact = '[ fs ]';
input_to_Wx1 = '[ qcar(1) ]';
qcaric1 = sysic; % 完成增广模型建立

% 控制器求解
ncont = 1;
nmeas = 1;
[K1, Scl1, gam1] = hinfsyn(qcaric1, nmeas, ncont);

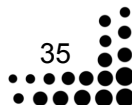
% 设计结果作图
CL1 = lft(qcar([1:4 3],1:2),K1); % 建立特定输入输出通道间闭环传递函数
bodemag(qcar(3,1),'k-.',CL1(3,1),'r-',logspace(0,2.3,140)) % 作特定闭环
传递函数伯德图

```

程序大致可分为：建立状态方程、建立增广模型、控制器求解和设计结果作图四部分。

1. 建立状态方程

建立状态方程分为变量赋值和状态方程组合两部分。其中状态方程组合用命令 `ss` 完





成，ss 函数的具体用法请见控制工程工具箱介绍。

2. 建立增广模型

建立增广模型利用命令 `sysic` 完成，这里建立的增广模型名为“`qcaric1`”。建立增广模型有以下两个重要的步骤。

(1) 定义状态变量、输入变量和输出变量，分别用 `Systemnames`，`Inputvar` 和 `Outputvar` 完成。

(2) 定义这些变量间关系。如 `input_to_qcar = '[Wref; fs]'` 表示输入到汽车悬架模型 `qcar` 的是 `Wref` 和 `fs` 这两个变量。

3. 控制器求解

控制器求解由命令 `hinfsvn` 完成。控制器求解需按照一定的求解算法，如果读者对这些算法感兴趣，可参考相关书籍。

4. 设计结果作图

设计结果作图由命令 `bodemag` 完成，该命令可作出系统传递函数的伯德图，显示设计结果的频域分析，为评估设计结果提供依据。其中用的命令 `bodemag` 和 `lft` 可参见控制工程工具箱。

根据程序可进行仿真，结果如图 3-7 所示。图中点划线为被动悬架系统(即没有控制)闭环传递函数频域图，实线表示主动悬架系统闭环传递函数频域图。由图可以看出，在相同的干扰作用下，主动悬架系统相对于被动悬架系统具有更小的扰动，即说明主动悬架系统能为乘客提供更舒适的乘坐环境。另一方面也说明设计控制器的有效性。

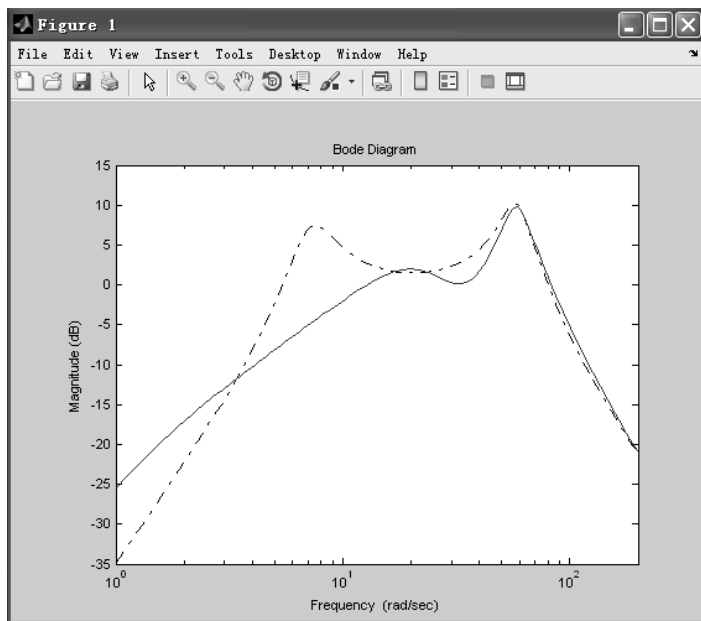


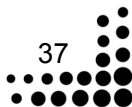
图 3-7 未考虑不确定性的汽车被动与主动悬架系统频域特性对比

3.3 本例小结

本例首先基于 Simulink 对汽车被动悬架系统进行建模仿真，然后基于鲁棒设计方法对汽车主动悬架系统进行控制器设计并进行建模仿真，通过对比展示主动悬架系统在动力学特性方面的优点。

通过本例的学习，读者应该掌握以下几个知识点：

- 进一步掌握基于 Simulink 和 M 语言进行系统建模仿真的过程与特点。
- 了解被动悬架系统与主动悬架系统不同点以及两者的动力学特性。
- 掌握基于鲁棒设计方法进行控制器设计的过程。





第 4 例 卫星对地定向姿态控制设计

这里的卫星指人造卫星，卫星指由人类建造，利用火箭、航天飞机等发射到太空中，像天然卫星一样环绕地球或其他天体运行的装置。自前苏联发射第一颗人造卫星起，人类已经发射了众多的卫星。这些卫星对人类的生活产生了重要的影响，如气象卫星能够提供更为准确的天气预报，资源卫星能够更为迅捷地发现各种资源，通信卫星能使通信变得更加畅通。对卫星及相关技术的研究一直是各国研究的热点。本例基于线性矩阵不等式 (LMI) 方法对卫星对地定向姿态运动设计控制器，并进行仿真。在设计之前，为使读者熟悉基于线性矩阵不等式方法设计过程，首先对其进行简单介绍，其次介绍卫星对地定向姿态运动动力学，并进行控制器设计，最后给出仿真程序。

通过本例学习应了解和掌握以下几方面：

- LMI 工具箱基本原理。
- 卫星对地定向动力学。
- 卫星对地定向控制器设计。

4.1 LMI 工具箱简介

近年来，线性矩阵不等式 (LMI) 广泛应用于解决系统与控制中的一系列问题。随着解决 LMI 内点法的提出以及 MATLAB 中 LMI 控制工具箱的推广，LMI 这一工具已经受到人们重视。

MATLAB 中的 LMI 控制工具箱，采用内点法对 LMI 进行求解，这些求解器比经典的凸优化算法速度有了显著提高。另一方面，它采用了有效的 LMI 结构化表示，为求解和计算领域做出了重大贡献。下面对 MATLAB 中的 LMI 控制工具箱予以介绍。

4.1.1 LMI 基本概念

LMI 具有的一般形式如式 (4-1) 所示：

$$F(x) = F_0 + x_1 F_1 + \cdots + x_m F_m < 0 \quad (4-1)$$

式中的小于号表示 $F(x)$ 是负定的。 $F_i = F_i^T \in \mathbf{R}^{n \times n}$ 是给定的对称矩阵， x_i 是待确定的变量，称为 LMI 系统的决策变量。

式 (4-1) 是 LMI 基本的形式，但在许多控制问题中，变量往往以矩阵的形式给出，例如 Lyapunov 矩阵不等式如式 (4-2) 所示：



$$A^T X + XA + Q < 0 \quad (4-2)$$

其中, A 和 Q 是给定的常数矩阵, 且 Q 是对称正定矩阵, X 是对称的未知矩阵变量。

这两种形式可以实现转换, 为方便使用, 在 MATLAB 中的 LMI 控制工具箱采用矩阵的形式, 即更符合控制问题一般形式的方式。

LMI 工具箱采用一般形式表示的 LMI 系统如式 (4-3) 所示:

$$N^T L(X_1, X_2, \dots, X_n) N < M^T R(X_1, X_2, \dots, X_n) M \quad (4-3)$$

式中, N 和 M 为外因子, $L(X_1, X_2, \dots, X_n)$ 和 $R(X_1, X_2, \dots, X_n)$ 为内因子。 $N^T L(X_1, X_2, \dots, X_n) N$ 称为左边, $M^T R(X_1, X_2, \dots, X_n) M$ 称为右边, $X_i, i=1, 2, \dots, n$ 为矩阵变量。

式 (4-3) 具有以下特征:

- 左边是不等式较小的一边, 右边则是较大的一边。例如 $A > B$, A 是不等式的右边, 而 B 是不等式的左边。
- 式中 N 和 M 具有相同维数, 可以不是方阵, 可以存在也可以不存在。都不存在时, 不等式化为: $L(X_1, X_2, \dots, X_n) < R(X_1, X_2, \dots, X_n)$, 当然也可以单独存在 N 或者 M 。
- L 和 R 是内因子, 它们是分块对称矩阵, 且两者具有相同块结构。由于是对称矩阵, 因此在 MATLAB 编程时, 只需写出一半即可。
- 内因子中的子块矩阵中每一个表达式都称为 LMI 的项, 它们是关于矩阵变量及其他标量的仿射表达式。

4.1.2 LMI 求解问题类型

LMI 求解问题可分为三类。

1. 可行性问题

寻找一个 $X \in R^n$, 使满足 LMI 系统成立, 这个问题在 MATLAB 中求解器的命令为: feasp。

2. 线性目标最小化问题

线性目标最小化问题如式 (4-4) 所示:

$$\begin{aligned} \min_x & c^T x \\ \text{s.t.} & A(x) < B(x) \end{aligned} \quad (4-4)$$

该问题是针对部分或者全部变量的线性目标进行优化求得对应变量值, 相应的求解器命令为: mincx。

3. 广义特征值最小化问题

广义特征值最小化问题如式 (4-5) 所示:



$$\begin{aligned} \min_x & \lambda \\ \text{s.t.} & C(x) < D(x) \\ & 0 < B(x) \\ & A(x) < \lambda B(x) \end{aligned} \quad (4-5)$$

该问题针对目标矩阵特征值最优情况下变量求解，对应的命令为：gevp。

4.1.3 LMI 建模求解函数

LMI 系统的描述既可以通过交互式 GUI 界面 `lmiedit` 来进行直观的矩阵描述，又可以通过命令 `lmivar` 和 `lmiterm` 来进行逐项的描述。

交互式函数 `lmiinfo` 提供有 `lmiedit`、`lmivar` 和 `lmiterm` 所创建 LMI 系统的量化检索功能。也可以通过 `lmiedit` 使得由一系列 `lmivar` 和 `lmiterm` 命令所建立的 LMI 系统可视化。

一般的 LMI 求解器可针对上面的三类标准的 LMI 问题进行求解，三个求解器可以解决一般结构的 LMI 系统和矩阵变量，并返回一个关于决策变量 x 的可行或者最优解，相应的矩阵变量可通过函数 `dec2mat` 得到。

所得解 x 可通过 `evallmi` 和 `showlmi` 进行验证，从而实现检查分析结果。LMI 系统中的所有变量都可以通过 `evallmi` 有决策变量 x 的值来估计，此时，每个 LMI 左边和右边的部分都变成了常数矩阵，可通过 `showlmi` 查看。

LMI 工具箱常用函数如表 4-1 所示。

表 4-1 LMI 工具箱常用函数

LMI 函数	说 明
<code>setlmi</code>	初始化 LMI 系统
<code>lmivar</code>	定义矩阵变量
<code>lmiterm</code>	确定 LMI 系统中每一项的内容
<code>newlmi</code>	多 LMI 系统中添加新的 LMI
<code>getlmi</code>	获得 LMI 系统的内部描述
<code>lmiedit</code>	通过 GUI 界面确定 LMI 系统
<code>dec2mat</code>	将求解器的输出转化为矩阵变量值
<code>mat2dec</code>	通过给定的矩阵变量值返回决策向量
<code>feasp</code>	验证 LMI 的可行性
<code>mincx</code>	LMI 限制下线性目标的极小值
<code>defcx</code>	在 <code>mincx</code> 命令中第一 $C^T x$ 目标
<code>gevp</code>	LMI 限制下的广义特征值最小化
<code>evallmi</code>	由决策变量的给定值来验证所有的变量项
<code>showlmi</code>	返回一个已经评估的 LMI 的左右边
<code>dellmi</code>	从系统中删除一个 LMI
<code>delmvar</code>	从问题中移除一个矩阵变量



续表

LMI 函数	说 明
setmvar	将一个矩阵变量赋予指定值
decinfo	以决策变量的形式表示每个输入的矩阵变量
decnbr	得到决策变量的个数
lmiinfo	查询现存 LMI 系统的信息
lmiinbr	得到问题中 LMI 的个数
mntnabr	得到问题中矩阵变量的个数

下面对部分函数进行详解。

1. setlmis([])或者 setlmis(lmi0)

在通过 lmiivar 以及 lmiterm 描述一个 LMI 系统之前,利用 setlmis 初始化其内部表示。为一个已经存在的 LMI 系统中添加新描述,使用后者,其中 lmi0 表示已存在的 LMI 系统,之后的 lmiivar 和 lmiterm 被用来在 lmi0 中添加新的变量和项。

2. [X, n, sX]=lmiivar(type,struct)

为 LMI 问题定义矩阵变量,其中 type 和 struct 是描述该矩阵变量的必须参数。type 确定变量 X 的类型,struct 描述变量 X 的内容。

type=1: 此时 X 是具有块对角化对称矩阵。 X 对角线上的每一个矩阵 D_i 必须是方阵,注意标量也是 1×1 的方阵。此时的矩阵变量 X 大体结果如下:

$$X = \begin{bmatrix} D_1 & & & \\ & D_2 & & \\ & & \ddots & \\ & & & D_n \end{bmatrix}$$

注意: $D_i, i=1,2,\dots$ 必须为方阵。

如果 X 具有 n 个对角块,那么 struct 是一个 $n \times 2$ 的矩阵:

$m = \text{struct}(i,1)$ 表示第 i 个方阵的大小,比如 D_i 是 5×5 的方阵,那么 $\text{struct}(i,1) = 5$ 。

$n = \text{struct}(i,2)$ 表示 D_i 的内容, $n = -1$ 表示 D_i 是零矩阵, $n = 0$ 表示标量, $n = 1$ 表示满矩阵。

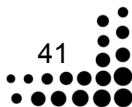
type=2: 表示 X 是一个 $m \times n$ 的长方形矩阵,此时 $\text{struct} = [m, n]$ 。

type=3: 表示其他结构,一般用于复杂的 LMI 系统,正常情况下使用较少。

3. lmiterm(termID,A,B,flag)

确定 LMI 中每一项的内容,包括内外因子、常数项以及变量项。

termID: 四元向量,确定该项位置及包含的矩阵变量。其中第一个元素表示描述的项属于哪个 LMI,它可取为 p 或 $-p$ (p 为正整数), p 代表该项位于第 p 个 LMI 的左边,而 $-p$ 则代表该项位于第 p 个 LMI 的右边。第二个和第三个元素组合在一起表示描述的项在特定 LMI 中矩阵块的位置,第二个元素为矩阵块中行位置,第三个元素为矩阵块中列位





置。第四个元素表示描述项是常数还是变量。如果为 0 则表明描述项是常数，如果为变量名则表明描述项是变量，此外如果变量名前为负号，则表示该变量的转置，如果为正则表示变量本身。

A 和 B 描述该项包含的数据信息，可以为常数或已知变量。分为两种情况：第一种是 termID 中第四个元素为 0，此时 B 省略，该项为常数 A；第二种情况是 termID 中第四个元素为变量，设该变量为 X，则此时该项为 AXB，即 A 和 B 分别左乘和右乘该变量。

flag 是可选参数，且只能为's'，它只对 $AXB+B^T X^T A$ 有效，即可以一次性添加这两项，具体命令为：lmiterm([1 1 1 X], A, B, 's')；

4. tag = newlmi

在当前描述的 LMI 系统中添加新的 LMI，并返回句柄 tag。

5. lmisys = getlmis

在利用 lmivar 和 lmiterm 描述给定 LMI 后，通过 getlmis 获取当前 LMI 系统内部描述。

4.2 卫星对地定向动力学模型

卫星对地定向动力学模型由式 (4-6) 和式 (4-7) 组成：

$$\begin{cases} J_x \dot{\omega}_x - (J_y - J_z) \omega_y \omega_z = M_x \\ J_y \dot{\omega}_y - (J_z - J_x) \omega_z \omega_x = M_y \\ J_z \dot{\omega}_z - (J_x - J_y) \omega_x \omega_y = M_z \end{cases} \quad (4-6)$$

式中 J_x, J_y, J_z 为卫星转动惯量， $\omega_x, \omega_y, \omega_z$ 为卫星角速度， M_x, M_y, M_z 为作用于卫星上的控制力矩。

$$\begin{cases} \dot{\omega}_x = \ddot{\phi} - \omega_0 \dot{\psi} \\ \dot{\omega}_y = \ddot{\theta} \\ \dot{\omega}_z = \ddot{\psi} + \omega_0 \dot{\phi} \end{cases} \quad (4-7)$$

式中 ϕ, θ, ψ 为卫星对地定向时的欧拉角。

卫星对地定向控制的目标是使欧拉角 ϕ, θ, ψ 为零。

为设计控制器可采用简化的模型。在小角度的前提下，可略去欧拉角之间的耦合，采用双积分的简化模型，即三个欧拉角动力学可简化为式 (4-8)：

$$\ddot{\alpha} = \frac{M_i}{J_i} + \Delta a_i \quad (\alpha = \phi, \theta, \psi \quad i = x, y, z) \quad (4-8)$$

式中 Δa_i 为简化模型与实际模型之间误差。

将式 (4-8) 写成状态方程形式如式 (4-9) 所示：



$$\begin{cases} \dot{x} = Ax + B_1 w + B_2 u \\ z_0 = C_0 x + D_0 u \\ z_1 = C_1 x + D_1 u \end{cases} \quad (4-9)$$

式中 $A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}$, $B_1 = B_2 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $C_0 = C_1 = [1 \quad 0]$, $D_0 = D_1 = 0$ 。

4.3 控制器设计及仿真

控制器采用混合 H2/H ∞ 方法进行设计。关于混合 H2/H ∞ 方法具体证明过程读者可参考相关文献,这里只给出结论。对于式(4-9),系统达到混合 H2/H ∞ 性能最优需满足等式(4-10)。若满足等式(4-10),则控制器 $K = WY^{-1}$ 。等式(4-10)由两个线性矩阵不等式组成,可利用前面介绍的 LMI 工具箱进行求解。

$$\begin{pmatrix} X & C_0 Y + D_0 W \\ (C_0 Y + D_0 W)^T & Y \end{pmatrix} > 0$$

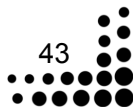
$$\begin{pmatrix} -AY - YA^T - B_2 W - W^T B_2^T - B_1 B_1^T & (C_1 Y + D_1 W)^T \\ (C_1 Y + D_1 W) & \gamma^2 I \end{pmatrix} > 0 \quad (4-10)$$

式中 X, Y 和 W 是需要优化的目标变量,其中 X, Y 为对称正定矩阵。

求解等式(4-10)及控制器的程序如下:

```
gama=2; % 鲁棒性能参数
A=[0 1; 0 0]; % 等式(4-9)中变量赋值
B1=[0 1]';
B2=[0 1]';
C0=[1 0];
D0=[0];
C1=[1 0];
D1=[0];

setlmis([]); % 开始建立 LMI
X = lmivar(1,[1 1]); % 为变量设置存储空间
W = lmivar(2,[1 2]); % lmivar 的用法可参见 4.1.3 小节介绍
Y = lmivar(1,[2 1]);
lmiterm([-1 1 1 X],1,1); % 建立线性矩阵不等式对应项
lmiterm([-1 1 2 Y],C0,1); % lmiterm 的用法可参见 4.1.3 小节介绍
lmiterm([-1 1 2 W],D0,1);
lmiterm([-1 2 1 -Y],1,C0');
lmiterm([-1 2 1 -W],1,D0');
lmiterm([-1 2 2 Y],10,10);
lmiterm([-2 1 1 Y],-A,1);
lmiterm([-2 1 1 Y],-1,A');
lmiterm([-2 1 1 W],-B2,1);
```





```

lmiterm([-2 1 1 -W],-1,B2');
lmiterm([-2 1 1 0],-B1*B1');
lmiterm([-2 1 2 -Y],1,C1');
lmiterm([-2 1 2 -W],1,D1');
lmiterm([-2 2 1 Y],C1,1);
lmiterm([-2 2 1 W],D1,1);
lmiterm([-2 2 2 0],gama*gama*eye(1));
LMIs = getlmis;           % 完成 LMI 建立
[tmin,xfeas] = feasp(LMIs) % 求解 LMI
X = dec2mat(LMIs,xfeas,X); % 取出变量值
W = dec2mat(LMIs,xfeas,W);
Y = dec2mat(LMIs,xfeas,Y);
K = W*inv(Y)              % 确定控制器

```

求解出控制器后,可对卫星姿态控制进行仿真,仿真由 M 文件和 Simulink 文件组成,其中仿真程序如下:

```

Jx = 2000; Jy = 1800; Jz = 250;
w0 = 2*pi/5400;
sim('satellite') % 利用 MATLAB 语言驱动 Simulink 文件仿真
subplot(3,1,1), plot(simout.time, simout.signals.values(:,1)*57.3),
ylabel('fai')
subplot(3,1,2), plot(simout.time, simout.signals.values(:,2)*57.3),
ylabel('zeta')
subplot(3,1,3), plot(simout.time, simout.signals.values(:,3)*57.3),
xlabel('t/sec'),ylabel('bosai')

```

说明: sim 命令能够驱动 Simulink 文件仿真,由于 Simulink 和 MATLAB 共享内存空间,因此 Simulink 文件仿真返回到内存空间的数据能够被 M 文件使用。上述程序后面的作图数据即来源于 Simulink 文件仿真。

Simulink 文件如图 4-1 至图 4-4 所示。其中图 4-1 和图 4-2 给出控制器框图,图 4-3 和图 4-4 给出动力学模型,对应式 (4-6) 和式 (4-7)。图 4-1 中模块“To Workspace”能够将仿真运行的数据保存到内存空间以备后用。这里就是利用其将卫星仿真过程中的欧拉角数据保存下来,再通过作图命令将结果显示出来。

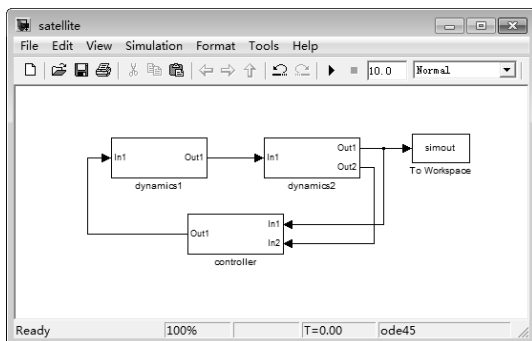


图 4-1 卫星对地观测姿态控制仿真框图

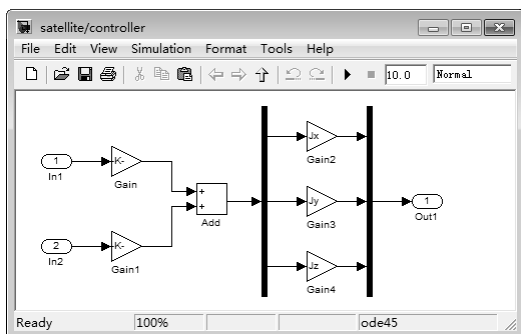


图 4-2 卫星对地观测姿态控制器

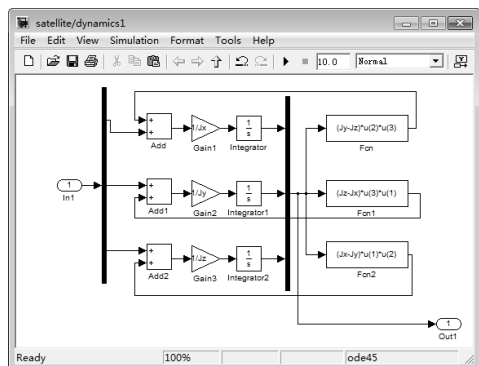


图 4-3 卫星对地观测姿态动力学模型

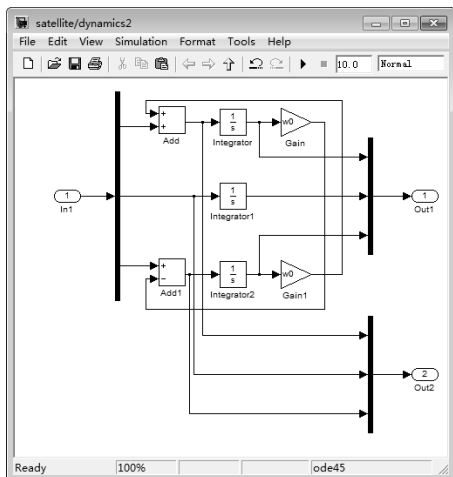


图 4-4 卫星对地观测姿态运动学模型

通过 M 文件和 Simulink 文件进行混合仿真得仿真结果如图 4-5 所示。由图可看出卫星三个欧拉轴都能迅速实现稳定，从而表明了控制器设计以及 LMI 工具箱的有效性。

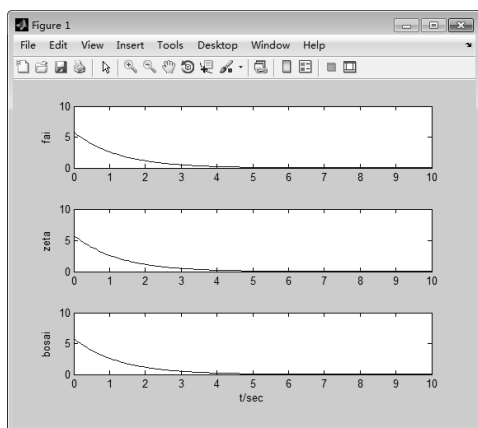


图 4-5 卫星对地观测姿态控制结果

4.4 本例小结

本例首先介绍了 MATLAB 的 LMI 工具箱基础知识。然后通过对地定向姿态控制实例来展示基于 LMI 工具箱进行仿真设计技巧以及基于混合 H_2/H_∞ 设计控制器的方法。

通过本例的学习，读者应该掌握以下几个知识点：

- 了解 LMI 工具箱基础知识。
- 掌握利用 LMI 工具箱进行 LMI 建立与求解过程。
- 掌握基于混合 H_2/H_∞ 设计控制器过程。
- 了解并掌握 MATLAB 语言和 Simulink 混合仿真方法。

第二部分 通信工程仿真实例



引言——通信系统分类及 MATLAB 通信工具箱简介（上）

一、通信系统分类

通信工程是信息科学技术发展的一个领域，分为模拟移动通信、数字移动通信和网络通信等，它以现代的声、光、电等技术为硬件基础，辅以相应软件来达到信息交流的目的。在多媒体广泛推广和互联网的应用前提下，通信工程获得了迅速的发展。

通信工程专业主要研究信号的产生、信息的传输、交换和处理，以及在计算机通信、数字通信、卫星通信、光纤通信、蜂窝通信、个人通信、平流程通信、多媒体技术、信息高速公路、数字程控交换等方面的理论和应用问题。从 19 世纪美国人发明电报开始，通信工程已经产生。

通信的目的是传递消息，按照不同的分法，通信可分成许多类别，下面介绍几种较常用的分类方法。

1. 按传输媒质

按消息由一地向另一地传递时传输媒质的不同，通信可分为两大类：一类称为有线通信，另一类称为无线通信。

2. 按信道中所传信号的特征

按照信道中传输的是模拟信号还是数字信号，可以相应地把通信系统分为模拟通信系统与数字通信系统。

3. 按工作频段

按通信设备的工作频率不同，通信系统可分为长波通信、中波通信、短波通信、微波通信等。

4. 按调制方式

根据是否采用调制，可将通信系统分为基带传输和频带（调制）传输。基带传输是



将没有经过调制的信号直接传送，如音频市内电话；频带传输是对各种信号调制后再送到信道中传输的总称。

5. 按业务的不同

按通信业务分，通信系统可分为话务通信和非话务通信两类。

6. 按通信者是否运动

通信还可按收发信者是否运动分为移动通信和固定通信。移动通信是指通信双方至少有一方在运动中进行信息交换。

从不同角度考虑问题，通信的工作方式通常有以下几种。

(1) 按消息传送的方向与时间分。对于点对点之间的通信，按消息传送的方向与时间，通信方式可分为单工通信、半双工通信及全双工通信 3 种。所谓单工通信，是指消息只能单方向进行传输的一种通信工作方式。所谓半双工通信方式，是指通信双方都能收发消息，但不能同时进行收和发的工作方式，对讲机、收发报机等都是这种通信方式。所谓全双工通信，是指通信双方可同时进行双向传输消息的工作方式，在这种方式下，双方都可同时进行收发消息。很明显，全双工通信的信道必须是双向信道，生活中全双工通信的例子非常多，如普通电话、手机等。

(2) 按数字信号排序方式分。在数字通信中，按照数字信号代码排列顺序的方式不同，可将通信方式分为串序传输和并序传输。

二、MATLAB 通信工具箱 (Communications Toolbox) 简介 (上)

在引言部分将对 MATLAB 通信工具箱中调制技术中的幅度调制和频率调制方法进行简单介绍，在第 5 例中将会对相位调制技术进行介绍。关于 MATLAB 通信工具箱中其他内容，由于篇幅限制，感兴趣的读者请参考 MATLAB 的帮助文档。

1. 幅度调制

通常作为信号发送的载波信号可以表示成：

$$s(t) = a(t) \sin(2\pi ft + \phi) \quad (\text{B1})$$

式中 $a(t)$ 表示幅度， f 是频率， ϕ 是相位，所以对载波信号可以改变的就是以上三个变量。根据改变变量的不同，可以分为幅度调制、频率调制和相位调制三大类型。

通过以上叙述可以知道，作为载波信号的是正弦信号，这是传统的载波信号。幅度调制常见的有以下几种。

1) BASK

从最简单的二进制幅度调制开始。设发送信号表示成：

$$s(t) = a_n g(t) \cos \omega t \quad (\text{B2})$$



这里 a_n 等于 0 或者 1。 $g(t) = \sum_n g(t - nT_s)$ 是基带信号波形，例如矩形波信号，或者通过脉冲成型滤波器后的输出信号。

假设在 1 秒内传送 10 个比特，那么代码如下：

```
% 仿真时间为 1 秒
t = 0 : 1/1e3 : 0.999;
% 十个随机数
a = randint(1, 10, 2);
% 十个方波
g = ones(1, 100);
g = [g, g, g, g, g, g, g, g, g, g];
% 二进制幅值调制
s = a(ceil(10*t+0.01)).*g.*cos(2*pi*100*t);
% 作图，结果如图 B1 所示
subplot(2, 1, 1);
plot(t, a(ceil(10*t+0.01)));
axis([0, 1, 0, 1.2]);
subplot(2, 1, 2);
plot(t, s);
```

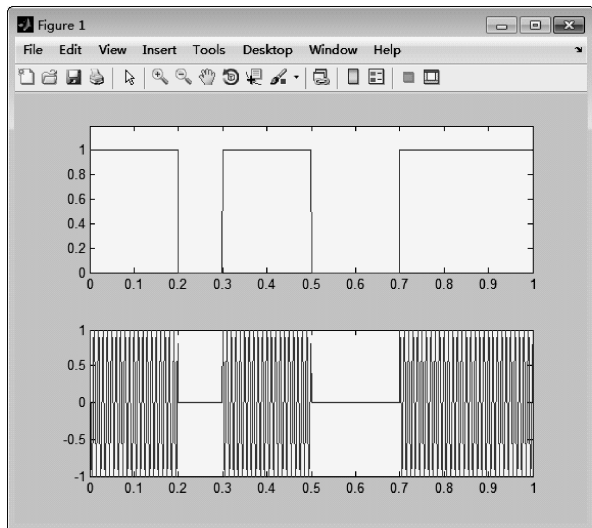


图 B1 BASK 调制图

2) MASK

以上分析了当输入是二进制比特流的情况，现在扩展成输入是 M 进制比特流的幅度调制。这里 $M = 2^N$ ， N 是大于或等于 2 的自然数，将 M 进制比特流映射成振幅值应该等于：

$$A = (2m - 1 - M)d \quad m = 1, \dots, M \quad (\text{B3})$$

式中 d 用来控制幅度间的差值，该值等于幅度间差值的一半。

下面的代码说明了一个四进制比特流的映射过程，其波形如图 B2 所示。



```

M = 4; d = 1;
t = 0 : 1/1e3 : 0.999;
a = randint(1, 10, M);
a = (2*a - 1 - M)*d;
g = ones(1, 100);
g = [g, g, g, g, g, g, g, g, g, g];
s = a(ceil(10*t+0.01)).*g.*cos(2*pi*100*t);
subplot(2, 1, 1);
plot(t, a(ceil(10*t+0.01)));
subplot(2, 1, 2);
plot(t, s);

```

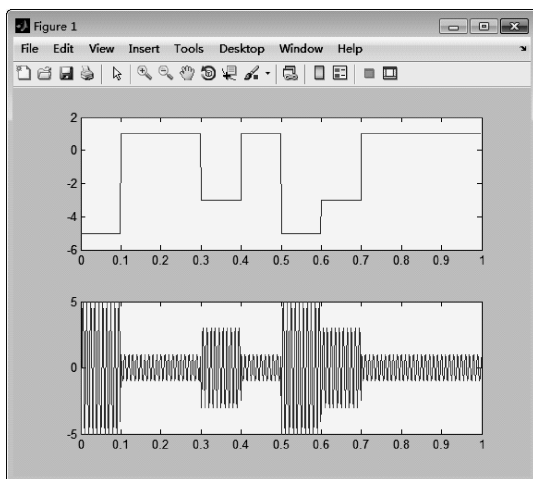


图 B2 MASK 调制图

如果 M 进制振幅调制的输入码元与二进制振幅调制的输入码元具有相同速率，那么 M 进制振幅调制与二进制振幅调制具有相同的带宽，而 M 进制振幅调制具有更高的带宽利用率，因为在单位时间内 M 进制振幅调制能传输更多的比特。虽然 M 进制振幅调制具有更高的带宽利用率，但是在相同信噪比下，随着 M 增大，误码率也增大，也就是因为功率受限。 M 进制振幅调制也可以使用单边带调制等方式。

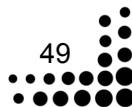
3) QAM

以上讨论都是单个载波的情况，也就是在单载波的情况下输入 M 进制比特流，现在考虑由两个载波分量的情况下输入 M 进制比特流，称这种调制为 QAM 调制方式。因为载波分量 $\cos \omega t$ 与 $\sin \omega t$ 是彼此正交的，也就是满足如下条件，在一个周期内有 $\int_0^T \cos \omega t \sin \omega t = 0$ ，所以该调制方式被称为正交幅度调制。其输出信号的表达式为：

$$s(t) = A_I g(t) \cos \omega t - A_Q g(t) \sin \omega t \quad (\text{B4})$$

式中 $A_I g(t) \cos \omega t$ 被称为同相分量， $A_Q g(t) \sin \omega t$ 被称为正交分量， A_I 、 A_Q 被称为同相载波分量幅度、正交载波分量幅度。

下面代码演示 QAM 调制过程，其调制图如图 B3 所示，其中 M 等于 4。





```

M = 4;
t = 0 : 1/1e3 : 0.999;
a = randint(10, 1, M);
Ai = 2*floor(a/2)' - 1;
Aq = 2*mod(a, 2)' - 1;
g = ones(1, 100);
g = [g, g, g, g, g, g, g, g, g, g];
s = Ai(ceil(10*t+0.01)).*g.*cos(2*pi*100*t) - Aq(ceil(10*t+0.01)).*g.*sin(2*pi*100*t);
plot(t, s);
scatterplot(Ai+j*Aq);

```

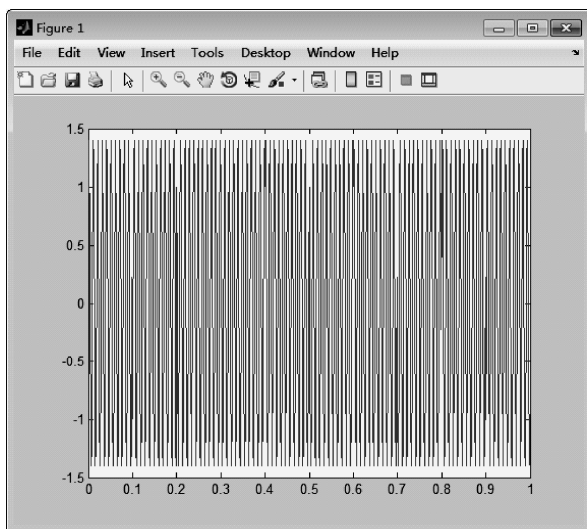


图 B3 QAM 调制图

2. 频率调制

1) BFSK

本节主要讨论频率调制技术，从最简单的二进制频率调制开始。设发送信号表示成：

$$s(t) = A \cos(2\pi(f + m\Delta f)t) \quad (B5)$$

式中 m 等于 ± 1 ， Δf 是相对于 f 的固定频率偏移。由于输入的比特流通常是由 0、1 组成，所以 $m = 2a - 1$ ，这里 a 是输入比特流。

下面代码生成 BFSK 信号并画出其如图 B4 所示的功率谱调制图。

```

%Time
t = 0 : 1/1e3 : 0.999;
%Frequency Offset
df = 10;
%Source & BFSK signal
a = randint(1, 10, 2);
m = 2*a(ceil(10*t+0.01))-1;

```



```
s = cos(2*pi*(100+m*df).*t);
%Calculate the Power Spectrum
f = 1000*(0:256)/512;
S = fft(s,512);
Pss = S.* conj(S) / 512;
subplot(3, 1, 1);
plot(t, a(ceil(10*t+0.01)));
axis([0, 1, 0, 1.2]);
subplot(3, 1, 2);
plot(t, s);
subplot(3, 1, 3);
plot(f, Pss(1:257));
```

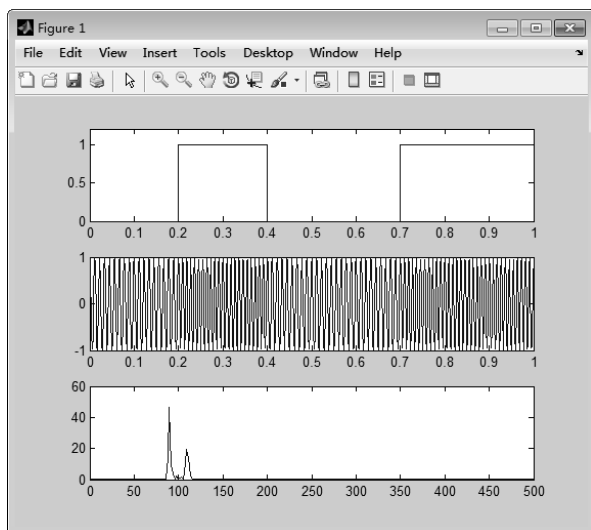
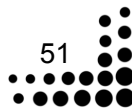


图 B4 BFSK 调制图

2) MFSK

前面讨论的是每次仅输入单个比特到调制器，如果每次输入多个比特到调制器，那么称为 M 进制频率调制，简称 MFSK。这里 $M=2^N$ ， $N \geq 1$ ， N 是每次输入到调制器的比特数。假设 N 等于 2，那么 M 等于 4，现在对四进制 FSK 编写如下代码，观察波形及其如图 B5 所示的功率谱调制图。

```
%Time
t = 0 : 1/1e3 : 0.999;
%Frequency offset & M = 4
df = 10; M = 4;
%Source & 4FSK
a = randint(1, 10, M);
m = 2*a(ceil(10*t+0.01))-3;
s = cos(2*pi*(100+m*df).*t);
%Power Spectrum
f = 1000*(0:256)/512;
```

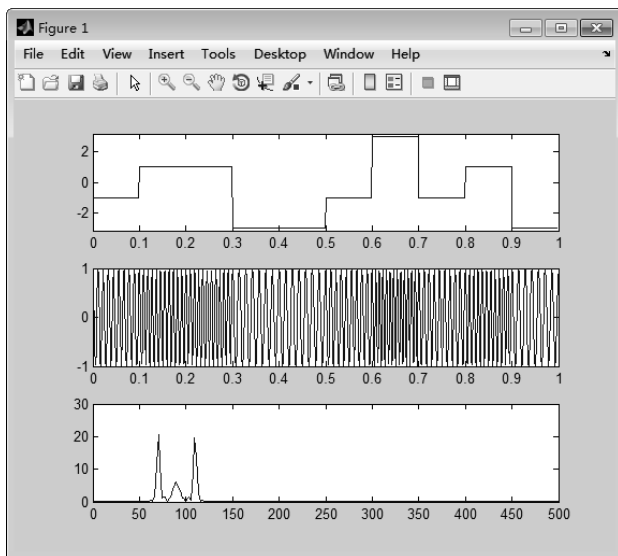




```

S = fft(s,512);
Pss = S.* conj(S) / 512;
%Drawing
subplot(3, 1, 1);
plot(t, m);
axis([0, 1, -3.2, 3.2]);
subplot(3, 1, 2);
plot(t, s);
subplot(3, 1, 3);
plot(f, Pss(1:257));

```



图B5 MFSK 调制图

3) CPFSK

以上两种频移键控调制方式，由于相位的非连续性导致谱扩展，有较大的频谱旁瓣产生。为避免产生较大的频谱旁瓣，可以将调制信号调制单一频率载波，并且相位连续变化，称这种频移键控为连续相位频移键控 CPFSK。其输出信号表达式为：

$$s(t) = \cos\left(2\pi f t + 4\pi T f_d \int_{-\infty}^t \sum_n I_n g(\tau - nT) d\tau\right) \quad (B6)$$

这里 T 是基带信号的周期， f_d 是峰值频率偏移。 I_n 是要发送的数字信息， $s(t)$ 是基带信号波形，通常可以采用矩形脉冲。虽然 $\sum_n I_n g(\tau - nT)$ 是不连续的，但是其积分却是连续的，所以该调制方式具有连续相位。

CPFSK 调制可以用如下代码表示，假设峰值频率偏移为 10Hz，基带信号周期为 0.1 秒，1 秒内传送 10 个比特。其调制图如图B6 所示。

```

%Frequency Offset and Singal Cycle
df = 10; T = 0.1;

```



```
%Duration
t = 0 : 1/1e3 : 0.999;
%Source and CPFSK output
a = 2*randint(1, 10, 2) -1;
m = cumsum(a);
a = a(ceil(10*t+0.01));
m = m(ceil(10*t+0.01));
s = cos(2*pi*100*t + 4*pi*T*df*cumtrapz(m)/1000);
%Power Spectrum
f = 1000*(0:256)/512;
S = fft(s,512);
Pss = S.* conj(S) / 512;
%Drawing
subplot(4, 1, 1);
plot(t, a);
axis([0, 1, -1.2, 1.2]);
subplot(4, 1, 2);
plot(t, m);
axis([0, 1, -3.2, 3.2]);
subplot(4, 1, 3);
plot(t, s);
subplot(4, 1, 4);
plot(f, Pss(1:257));
```

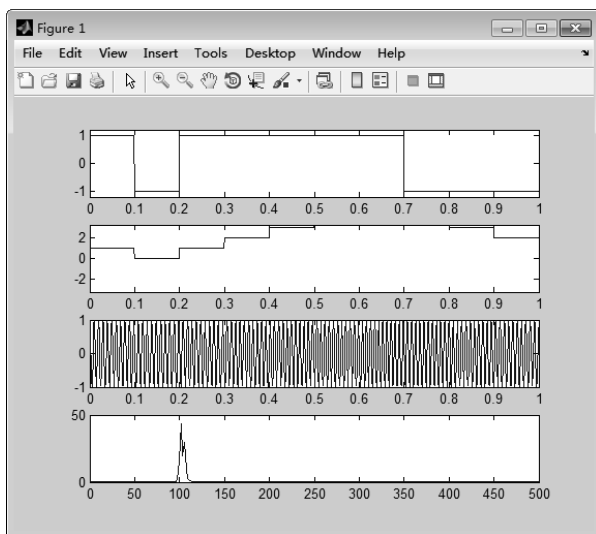
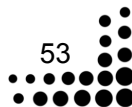


图 B6 CPFSK 调制图

4) MSK

MSK 是 CPFSK 的特例，这时调制指数 h 等于 $1/2$ ，即 $h = 2 f_d T = 1/2$ 。

将 CPFSK 的代码的 df 设为 2.5，这样 $2*f_d*T=0.5$ ，符合 MSK 的特性。生成的波形图如图 B7 所示。



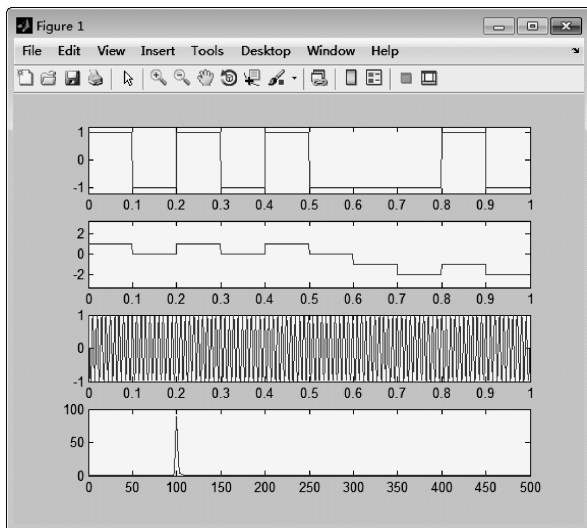


图 B7 MSK 调制图

如果在 MSK 调制器前加上高斯滤波器对输入信号滤波,那么调制器产生的输出信号被称为高斯最小频移键控 (GMSK)。

5) OFDM

OFDM 是正交频分复用 (Orthogonal Frequency Division Multiplex) 的缩写,是一种多载波频率调制方式。该调制方式利用一组正交信号作为载波分量,典型的正交信号如 $\{1, \cos \omega t, \cos 2\omega t, \dots, \cos m\omega t, \sin \omega t, \sin 2\omega t, \dots, \sin m\omega t\}$ 。其子载波通常采用 PSK 或 QAM 调制方式,这里假设子载波调制采用 QAM 调制方式。输出信号可以表示为:

$$s(t) = \sum_{m=0}^{M-1} [A_I g(t) \cos \omega_m t - A_Q g(t) \sin \omega_m t] \quad (B7)$$

M 是子载波个数, ω_m 是载波信号角频率, $\omega_m = 2\pi(f + m\Delta f)$, 这里 f 是 $1/T$ 的整数倍, Δf 等于 $1/T$, T 是基带信号周期。其代码如下,生成波形如图 B8 所示。

```
%Duration
t = 0 : 1/1e3 : 0.999;
%Source
m = 2*randint(20, 1, 2)-1;
a = m(1:4:20)';
b = m(2:4:20)';
c = m(3:4:20)';
d = m(4:4:20)';
m = m(ceil((1000*t+0.01)/50));
a = a(ceil((1000*t+1)/200));
b = b(ceil((1000*t+1)/200));
c = c(ceil((1000*t+1)/200));
d = d(ceil((1000*t+1)/200));
%OFDM Singal
```

```
s = a.*cos(2*pi*100*t) - b.*sin(2*pi*100*t) ...
+ c.*cos(2*pi*110*t) - d.*sin(2*pi*110*t);
%Power Spectrum
f = 1000*(0:256)/512;
S = fft(s,512);
Pss = S.* conj(S) / 512;
%Drawing
subplot(4, 1, 1);
plot(t, m);
axis([0, 1, -1.2, 1.2]);
subplot(4, 1, 2);
plot(t, a);
axis([0, 1, -1.2, 1.2]);
subplot(4, 1, 3);
plot(t, s);
subplot(4, 1, 4);
plot(f, Pss(1:257));
```

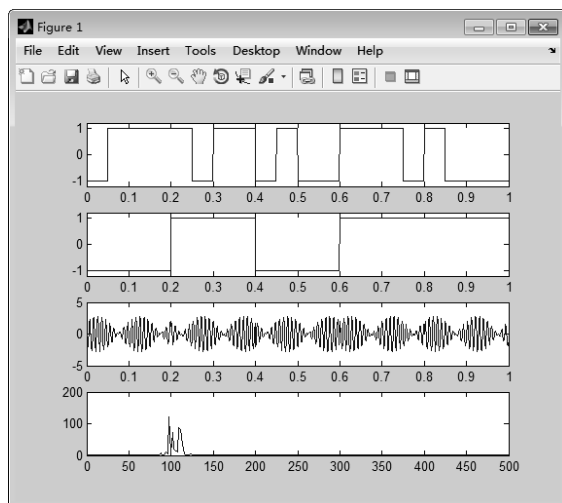


图 B8 OFDM 调制图

从图中可以看到 T 等于 0.2 秒，OFDM 信号幅度不是恒定的，另外在 100Hz、110Hz 处都有峰值。



第 5 例 车载数字电视调制解调设计

随着汽车在人们日常生活中得到越来越广泛的使用以及通信技术的进步，车载数字电视相关技术和设备已日趋成熟。本例对车载数字电视调制解调技术进行介绍。在介绍之前，首先对相位调制技术进行介绍。

通过本例学习，需了解和掌握以下几个方面：

- 了解和掌握相位调制技术。
- 了解车载数字电视调制解调技术。
- 掌握基于 MATLAB 通信工具箱设计和分析调制解调过程。

5.1 MATLAB 通信工具箱简介（下）

这里讨论相位调制技术，主要分为以下几种。

1. PSK

设发送信号表示成：

$$s(t) = g(t) \cos\left(\omega t + \frac{2\pi}{M}(m-1)\right) \quad m = 0, \dots, M-1 \quad (5-1)$$

这里 $g(t)$ 是基带信号波形。 M 是可能的相位个数， $M=2^N$ ， N 是每次被输入到调制器的比特数。当 M 等于 2，这时 PSK 通常被称为二进制相移键控 (BPSK)，当 M 等于 4，这时 PSK 通常被称为四进制相移键控 (QPSK)。如果输入的比特流先经过差分编码然后输入到调制器，这时 PSK 通常被称为差分相移键控 (DPSK)。

下面的代码采用 BPSK 调制方式，调制图如图 5-1 所示；如果设 M 等于 4，就是 QPSK 调制。

```
M = 2;
t = 0:0.001:0.999;
m = randint(10, 1, M)';
m = m(ceil(10*t+0.01));
s = cos(2*pi*100*t+m/M);
f = 1000*(0.256)/512;
S = fft(s,512);
Pss = S.* conj(S) / 512;
```




```
subplot(3, 1, 1);
plot(t, m);
axis([0 1 min(m)-0.2 max(m)+0.2]);
subplot(3, 1, 2);
plot(t, s);
subplot(3, 1, 3);
plot(f, Pss(1:257));
```

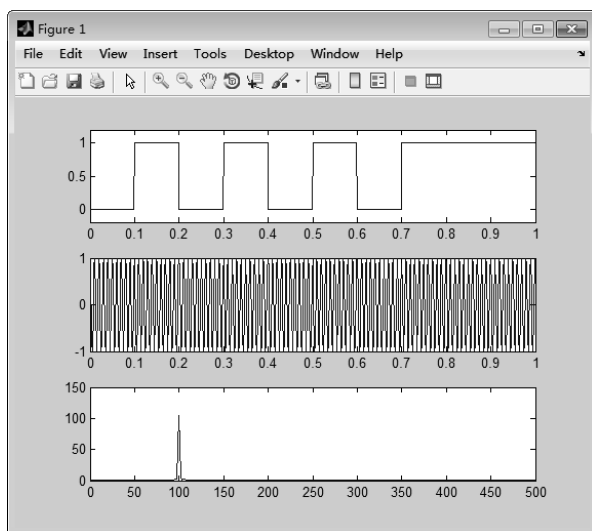


图 5-1 BPSK 调制图

2. OQPSK

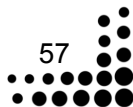
前面讨论的 FSK 以及现在讨论的 PSK 调制方式都是恒包络调制方式，这类调制方式可以在接收端采用限幅的方式来消除干扰引起的幅度变化，从而获得更好的抗干扰性能。

假定 QPSK 调制的基带波形是矩形波形，那么在频域调制后的信号具有无限带宽。通常信道都是带限的，当调制信号通过该信道后，输出信号不再是恒包络的。当 QPSK 调制在相邻码元间发生 180 度相移时，带限后的包络甚至会出现包络为 0 的现象。通过非线性带限信道后，在接收机由于功放的非线性，包络的起伏虽然可以被消减，但同时会使频谱扩展，从而对临近信道的信号形成干扰。

为避免出现 180 度的相移，可以采用 OQPSK (Offset QPSK) 调制方式。该调制方式首先串并变换，将输入比特流分成 I、Q 两路比特流，原先周期为 T_s 的一个比特流变成周期为 $2T_s$ 的两个比特流，然后将 Q 路比特流偏移一个周期 T_s ，按 QPSK 方式调制。这样由于在每个周期 T_s 只有某一路比特发生变化，而不会出现 I、Q 两路比特同时变化，所以就可以避免出现 180 度的相移。当 I、Q 两路比特同时变化，会出现 180 度的相移。

该调制方式发送信号可以表示成：

$$s(t) = I(t) \cos(2\pi ft) + Q(t - T_s) \sin(2\pi ft) \quad (5-2)$$





下面的代码采用 OQPSK 调制方式，调制图如图 5-2 所示。

```
%Duration
t = 0 : 1/1e3 : 0.999;
%Source
m = 2*randint(20, 1, 2) '-1;
I = m(1:2:20);
Q = m(2:2:20);
Q = [0, Q(1:end-1)];
m = m(ceil((1000*t+1)/50));
I = I(ceil((1000*t+1)/100));
Q = Q(ceil((1000*t+1)/100));
%QPSK
s = I.*cos(2*pi*100*t) - Q.*sin(2*pi*100*t);
%Power Spectrum
f = 1000*(0:256)/512;
S = fft(s,512);
Pss = S.* conj(S) / 512;
%Drawing
subplot(4, 1, 1);
plot(t, m);
axis([0, 1, -1.2, 1.2]);
subplot(4, 1, 2);
plot(t, I);
axis([0, 1, -1.2, 1.2]);
subplot(4, 1, 3);
plot(t, s);
subplot(4, 1, 4);
plot(f, Pss(1:257));
```

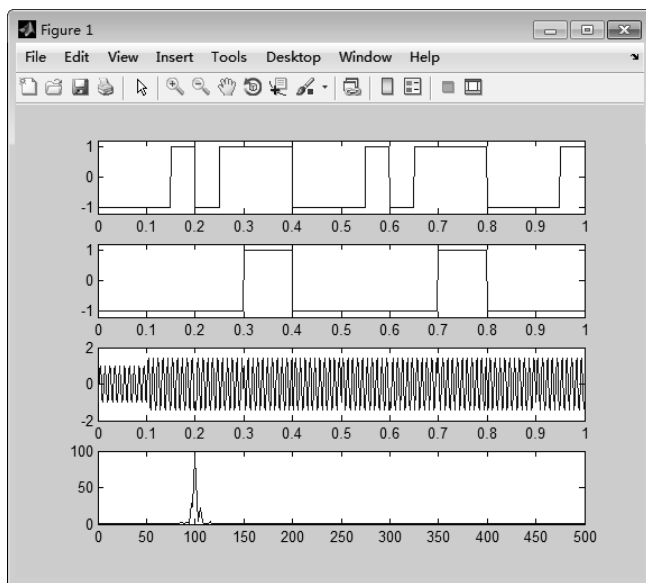


图 5-2 OQPSK 调制图



从图 5-2 中可以看到 OQPSK 调制方式占用较大的带宽。

由于相位的跳变导致通过带限信道后包络发生变化，所以如果采用连续相位变化就可以尽量避免因为相位跳变导致的包络变化。这就是连续相位调制，在频率调制已讨论过连续相位调制方式，例如 CPFSK、MSK 调制方式。

5.2 车载数字电视调制解调设计

设计调制解调分为以下几步。

1. 初始化

首先初始化变量和仿真参数，例如载频、采样率和噪声特性等。

```
% 初始化参数
Fc = 2.5e6;
Rsym = 1e6;
nSamps = 8;
frameLength = 2048;
M = 16;
EbNo = 15;
Fs = Rsym * nSamps;
SNR = EbNo + 10*log10(log2(M)/nSamps) + 10*log10(2);
```

初始化散点图和频谱估计等测量工具。

```
% 初始化参数点
specScope = spectrum.welch('Hamming',512);
specOpts=psdopts(specScope);
set(specOpts, 'Fs', Fs, 'SpectrumType','twosided','CenterDC',true);
scatScope = commscope.ScatterPlot('SamplingFrequency', Rsym, ...
    'SamplesPerSymbol', 1); close(scatScope)
```

2. 基频调制

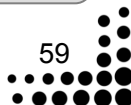
生成随机数，然后采用 16-QAM 调制方式。

```
% 生成 16-QAM 解调器
hMod = modem.qammod('M',M);
scatScope.Constellation = hMod.Constellation;
scatScope.PlotSettings.Constellation = 'on';
scatScope.PlotSettings.ConstellationStyle = '*r';
b = randi([0 hMod.M-1], frameLength, 1);
txSym = modulate(hMod, b);
```

3. 脉冲整形

调制后，利用余弦滤波器进行脉冲整形。通过频谱估计来验证基带信号的频谱是否集中在零赫兹附近。基频信号频谱估计结果如图 5-3 所示。

```
nSym = 8;          % Length of the filter in symbols
beta = 0.2;        % Rolloff factor
```





```
filterSpec = fdesign.pulseshaping(nSamps, 'Square root raised cosine', ...  
    'Nsym,Beta', nSym, beta);  
hXmtFlt = design(filterSpec);  
x = filter(hXmtFlt, upsample(txSym, nSamps));  
psdProbe = psd(specScope, x, specOpts);  
hFig = figure; plot(psdProbe)
```

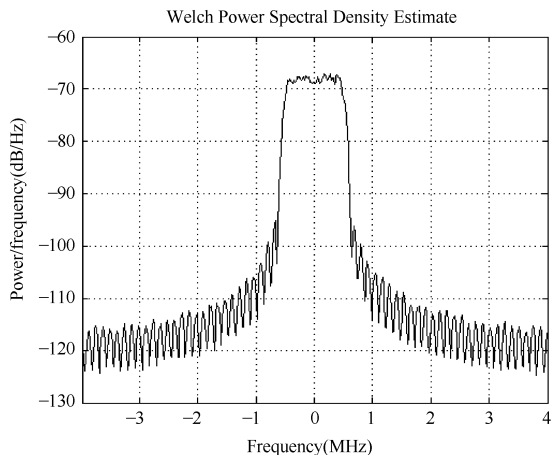


图 5-3 基频信号频谱估计结果

4. 频率上转换

通过将正弦信号与基频信号相乘来完成频率上转换过程。频率上转换频谱估计结果如图 5-4 所示。

```
t = (0:1/Fs:(frameLength/Rsym)-1/Fs).';  
carrier = sqrt(2)*exp(1i*2*pi*Fc*t);  
xUp = real(x.*carrier);  
  
% 估计频谱  
psdProbe = psd(specScope, xUp, specOpts);  
plot(psdProbe)
```

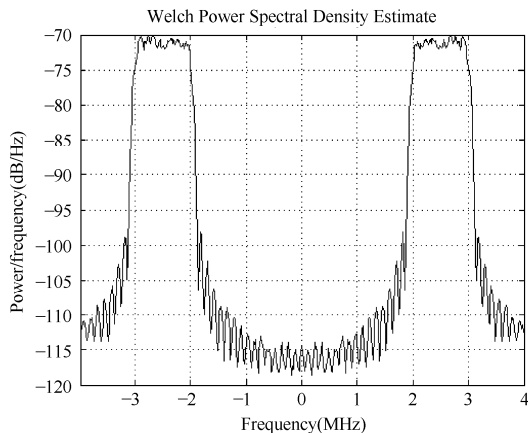


图 5-4 频率上转换频谱估计结果

5. 信道模拟

将信道模拟为带通 AWGN 信道，且由于立方非线性存在引起临近信道干涉。信道模拟结果如图 5-5 所示。

```
Fint = Fc/3+50e3;
interference = 0.7*cos(2*pi*Fint*t+pi/8).^3;
avgPwrBaseBand16QAM = 10*log10(sum(abs(hMod.Constellation).^2)/(M*nSamps));
sigPower = 10*log10(sum(hXmtFlt.Numerator.^2)) + avgPwrBaseBand16QAM;
yUp = awgn(xUp, SNR, sigPower);
yUpACI = yUp + interference;
psdProbe = psd(specScope, yUpACI, specOpts);
figure(hFig); hold on; hLine = plot(psdProbe); set(hLine, 'Color', [1 0 0]);
legend('Signal at channel input', 'Signal at channel output', 'Location',
'southwest')
```

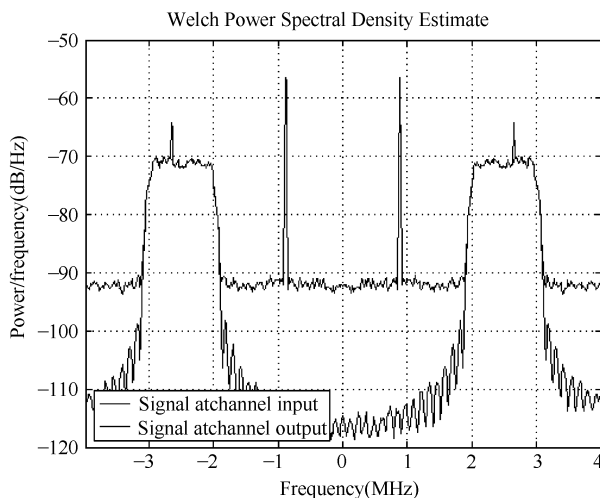


图 5-5 信道模拟结果

6. 频率下转换

通过将复基频信号与复正弦信号相乘进行频率下转换，频率下转换后得到解调前的基频。频率下转换结果如图 5-6 所示。

```
yACI = yUpACI.*conj(carrier);
psdProbe = psd(specScope, yACI, specOpts);
figure(hFig); hold off; plot(psdProbe);
```

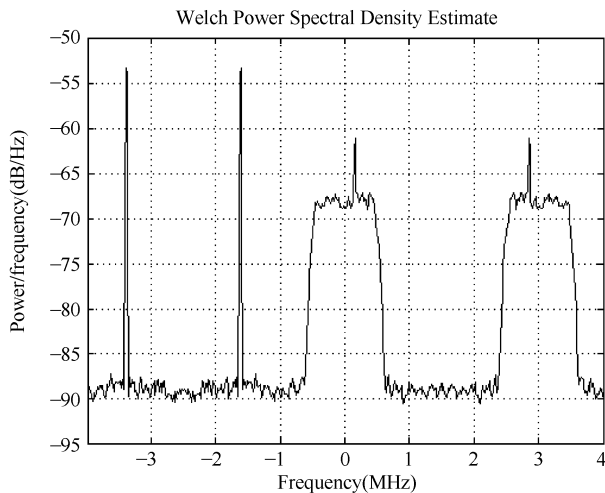


图 5-6 频率下转换结果

7. 匹配滤波

匹配滤波结果如图 5-7 所示。

```
hRcvFlt = copy(hXmtFlt);  
rcvSymACI = filter(hRcvFlt, yACI);  
psdProbe = psd(specScope, rcvSymACI, specOpts);  
figure(hFig); hold on; hLine = plot(psdProbe); set(hLine, 'Color', [1 0 0]);  
legend('Signal at filter input', 'Signal at filter output', 'Location',  
    'southwest')  
rcvSymACI = nSym*rcvSymACI;
```

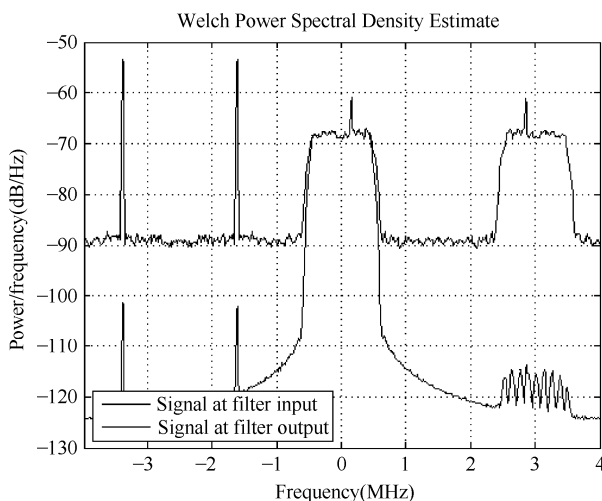


图 5-7 匹配滤波结果

8. 解调

匹配滤波之后，通过解调得到 16-QAM 信号，如图 5-8 所示。

```
e2eDelay = nSamps*nSym;
rcvSymDownACI = rcvSymACI(e2eDelay+1:nSamps:end);
update(scatScope, rcvSymDownACI)
hDemod = modem.qamdemod(hMod);
xHatACI = demodulate(hDemod, rcvSymDownACI);
numErr = sum(xHatACI~=b(1:end-nSym))
```

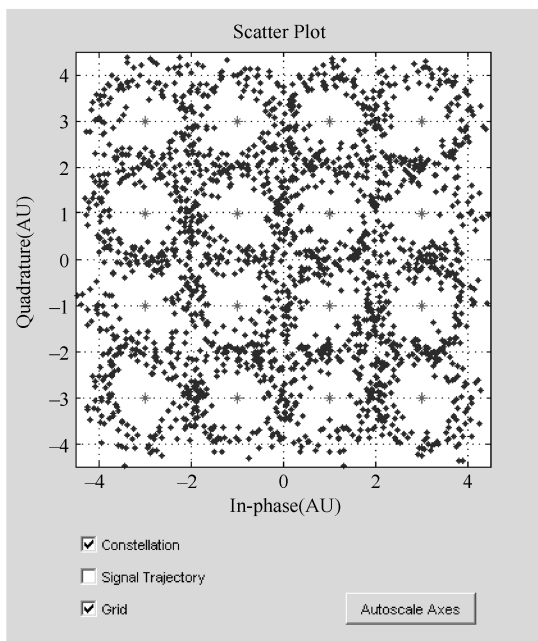


图 5-8 解调后得到的 16-QAM 信号

5.3 本例小结

本例以车载数字电视为对象，基于 MATLAB 通信工具箱模拟其调制解调过程，通过该例读者可以了解和掌握基于 MATLAB 通信工具箱进行通信系统建模、仿真的特点及流程，对于车载数字电视调制解调过程也将有所了解。



第 6 例 舰载雷达通信系统仿真

雷达用于探测近地或深空等肉眼不易观察的目标，以其捕获的准确迅速等特点在国防、深空探测等领域得到广泛应用。本例以舰载雷达为对象，研究其通信系统，为扩展 Simulink 应用，增加对 S 函数的介绍。

通过本例学习，需要了解和掌握以下几点：

- 了解 Simulink 仿真特点及流程。
- 掌握基于 M 语言的 S 函数编写。
- 了解雷达通信系统结构。
- 掌握基于 Simulink 对通信系统进行建模仿真的方法。

6.1 S 函数简介

Simulink 为建模仿真提供了良好的平台，用户可利用 Simulink 自带的模块方便快捷地建立所需的仿真模型。但是 Simulink 自带的模块是有限的，有时候不能找到所需的模块，这个时候可以通过编写 S 函数来进一步扩展 Simulink 的应用领域，以更好地完成建模与仿真任务。

函数可由 MATLAB 的 M 语言、C 语言、C++ 语言或 Fortran 语言编写。编写完之后通过将文件与 Simulink 提供的自定义 S 函数模块联系起来，就可在 Simulink 中正常使用了。

S 函数主要应用于以下几种情况：

- 创建一个新的模块。
- 表示硬件设备驱动。
- 需要重用 C 代码。
- 将系统描述为一系列数学方程。
- 需要用到图形动画。

Simulink 中运行分为多个步骤。首先是初始化，在这个阶段中，Simulink 解算器将确定各个模块的信号维数及宽度，数据类型、采样时间、模块参数和模块解算阶数，并分配内存空间。初始化结束后，Simulink 进入仿真循环，仿真每递进一步称为一个仿真步长。在每一个步长中，Simulink 都计算各个 S 函数及其他模块对应的状态、输入值微分值及输出值。如果模型包含连续状态，则存在内循环。解算器持续解算直到状态变量达到预定精度。Simulink 就是这样一个步长接着一个步长地循环计算直到仿真结束。



S 函数中包含一系列的回调函数,用于在 Simulink 仿真中完成每一个仿真步长中的任务。仿真的不同阶段都有对应的回调函数来完成相应的功能,这些回调函数有以下几种。

(1) 初始化回调函数。当仿真进行初始化时,解算器将 S 函数初始化,包括以下几方面:

- 初始化结构体“simStruct”,这个结构体包含了 S 函数的信息。
- 设置输入和输出端口的维数。
- 设置模块采样时刻。
- 分配存储空间。

(2) 计算下一次采样时刻。如果采用变步长采样模块,则需要首先计算出下一仿真步长的大小。

(3) 计算主要步长时刻对应的输出,当这一步完成后,当前输出可以为其他变量所用。

(4) 更新主要步长时刻对应的离散状态。

(5) 积分。如果 S 函数存在连续状态,则解算器需要在每一步长之间进行更小步长的迭代以保证足够的精度。

在 MATLAB 命令行中输入“edit sfuntmpl”这是 MATLAB 自己提供的 S 函数模板,具体分析 S 函数的结构,它的第一行是这样的: function [sys,x0,str,ts] = sfuntmpl(t,x,u,flag)。

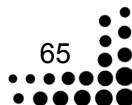
其中输入与输出变量的含义:t 是采样时间,x 是状态变量,u 是输入(是做成 simulink 模块的输入),flag 是仿真过程中的状态标志(以它来判断当前是初始化还是运行等);sys 输出根据 flag 的不同而不同(下面将结合 flag 来讲 sys 的含义),x0 是状态变量的初始值, str 是保留参数(一般在初始化中将它置空就可以了, str=[]), ts 是一个 1×2 的向量, ts(1) 是采样周期, ts(2) 是偏移量。

下面结合 sfuntmpl.m 中的代码来讲具体的结构:

```
switch flag, %判断 flag, 看当前处于哪个状态
case 0,
    [sys,x0,str,ts]=mdlInitializeSizes;
```

flag=0 表示处于初始化状态,此时用函数 mdlInitializeSizes 进行初始化,此函数在 sfuntmpl.m 的 149 行,在初始化状态下,sys 是一个结构体,用它来设置模块的一些参数,各个参数详细说明如下:

```
size = simsizes;%用于设置模块参数的结构体用 simsizes 来生成
sizes.NumContStates = 0;%模块连续状态变量的个数
sizes.NumDiscStates = 0;%模块离散状态变量的个数
sizes.NumOutputs = 0;%模块输出变量的个数
sizes.NumInputs = 0;%模块输入变量的个数
sizes.DirFeedthrough = 1;%模块是否存在直接贯通(直接贯通就是输入能直接控制
输出)
sizes.NumSampleTimes = 1;%模块的采样时间个数,至少是一个
sys = simsizes(sizes); %设置完后赋给 sys 输出
```





举个例子，考虑如下模型。

$dx/dt=fc(t,x,u)$ 也可以用连续状态方程描述： $dx/dt=A*x+B*u$ 。

$x(k+1)=fd(t,x,u)$ 也可以用离散状态方程描述： $x(k+1)=H*x(k)+G*u(k)$ 。

$y=fo(t,x,u)$ 也可以用输出状态方程描述： $y=C*x+D*u$ 。

设上述模型连续状态变量、离散状态变量、输入变量、输出变量均为 1 个，就只需改上面那一段代码为：

```
sizes.NumContStates=1;sizes.NumDiscStates=1;sizes.NumOutputs=1;sizes.
NumInpu
ts=1;
```

其他的可以不变。继续在 mdlInitializeSizes 函数中往下看：

```
x0 = []; %状态变量设置为空，表示没有状态变量，以上面的假设，可改为
x0=[0,0] (离散和连续的状态变量都设其初值为 0)
str = []; %这个就不用说了，保留参数，置[]即可
ts = [0 0]; %采样周期设为 0 表示是连续系统，如果是离散系统，在下面的 mdlGetTime
OfNextVarHit 函数中具体介绍
```

在 sfuntmpl 的 106 行继续往下看：

```
case 1,
sys=mdlDerivatives(t,x,u);
```

flag=1 表示此时要计算连续状态的微分，即上面提到的 $dx/dt=fc(t,x,u)$ 中的 dx/dt ，找到 mdlDerivatives 函数(在 193 行)如果设置连续状态变量个数为 0，此处只需 sys=[]; 就可以了(如 sfuntmpl 中一样)，按上述讨论的那个模型，此处改成 sys=fc(t,x(1),u)或 sys=A*x(1)+B*u %。这儿 x(1)是连续状态变量，而 x(2)是离散的，这里只用到连续的，此时的输出 sys 就是微分。

继续，在 sfuntmpl 的 112 行：

```
case 2,
sys=mdlUpdate(t,x,u);
```

flag=2 表示此时要计算下一个离散状态，即上面提到的 $x(k+1)=fd(t,x,u)$ ，找到 mdlUpdate 函数(在 206 行)，它这儿 sys=[];表示没有离散状态，这里可以改成 sys=fd(t,x(2),u)或 sys=H*x(2)+G*u;%sys，即为 x(k+1)。

在 sfuntmpl 的 118 行：

```
case 3,
sys=mdlOutputs(t,x,u);
```

flag=3 表示此时要计算输出，找到 mdlOutputs 函数(在 218 行)，如上，如果 sys=[]表示没有输出，改成 sys=fo(t,x,u)或 sys=C*x+D*u %sys，此时为输出 y。

在 sfuntmpl 的 124 行：

```
case 4,
sys=mdlGetTimeOfNextVarHit(t,x,u);
```

flag=4 表示此时要计算下一次采样的时间，只在离散采样系统中有用(即上文的

mdlInitializeSizes 中提到的 ts 设置 ts(1)不为 0)

连续系统中只需在 mdlGetTimeOfNextVarHit 函数中写上 sys=[];这个函数主要用于变步长的设置，具体实现可以用“edit vsfunc”。看 vsfunc.m 这个例子。

在 sfuntmpl 的 130 行：

```
case 9,
    sys=mdlTerminate(t,x,u);
```

flag=9 表示此时系统要结束，一般来说在 mdlTerminate 函数中写上 sys=[]就可，如果在结束时还要设置什么，就在此函数中写。

以上就是 sfuntmpl 这个 S 函数的模板。

6.2 舰载雷达通信系统建模仿真

舰载雷达通信系统框图如图 6-1 所示。由图可知雷达系统主要分为发射部分和接收部分。雷达系统首先发射信号至观测物体，物体反射该信号后返回雷达，雷达接收部分将该信号进行增益和滤波等操作后，将表征观测物体信息的数据显示出来。

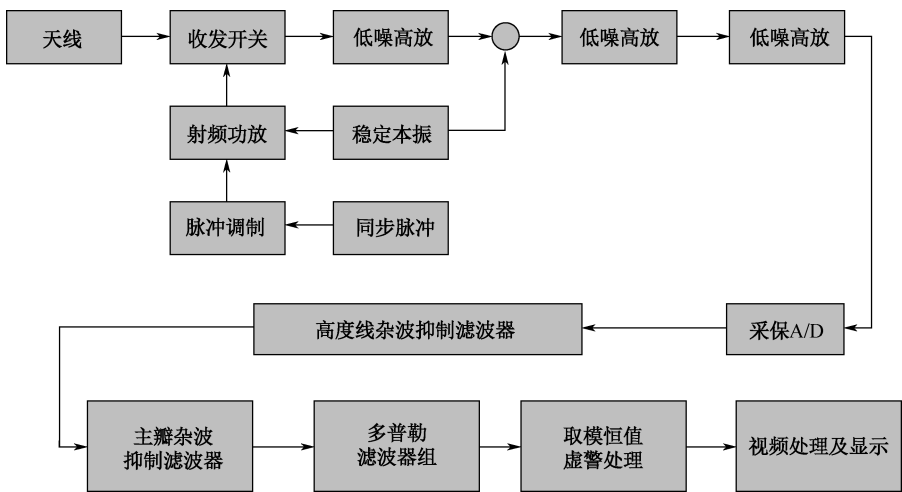


图 6-1 雷达系统结构框图

下面对雷达系统中的主要模型分别予以介绍。

1. 发射信号
- 发射机发射的单个脉冲信号可以表示成：
- $$P(t) = \sqrt{2P_t} A(t) \cos[2\pi f_c t + \theta(t)] \quad |t| \leq \frac{T_p}{2} \quad (6-1)$$
- 式中 P_t 表示发射机的峰值功率； f_c 为雷达发射机载频； T_p 为脉冲宽度； $A(t)$ 为脉冲幅度。



2. 目标反射信号

雷达的作用距离是目标横截面积的函数，在计算雷达作用距离时，往往按给定的横截面积进行计算，但实际雷达目标在空中由于受多种因素的影响，使其等效横截面积不是一个常量，而是一个随机变量，称之为目标的起伏，这种目标称为起伏目标。如果目标起伏很小，可忽略，就称之为非起伏目标。

若想准确估计目标横截面积的起伏，须知道概率密度函数。由于目标是千差万别的，因此想要得到完整的数据比较困难。但从目标起伏特性的研究表明，用一些模型对各类目标的起伏统计特性进行逼近是合理的。通常用功率信噪比 S 来研究雷达横截面积。

当前，采用的模型可分为两大类：经典的目标模型和现代的目标模型。前者主要包括恒指的马克姆（Marcum）模型和斯威林（Swering）起伏模型。

现代模型主要包括 χ^2 模型，莱斯模型和对数-正态模型等。

经典目标模型与许多实际目标非常接近，但具有一定的局限性，而现代模型具有更好的一般性。

一个匀速运动的起伏点目标反射信号可表示为：

$$S(t) = \operatorname{Re} \left[\tilde{S}(t) e^{j2\pi f_c t} \right] \left| t - \frac{2R}{c} \right| \frac{T_p}{2} \quad (6-2)$$

式中 $\tilde{S}(t) = \tilde{K} \tilde{b} \tilde{p} \left(t - \frac{2R}{c} \right) e^{j2\pi f_c t}$ ， $\tilde{K} = \sqrt{\frac{P_t L_s}{(4\pi)^3}} \frac{\tilde{G}_t \tilde{G}_r \lambda}{R^2}$ 为雷达距离方程常数， L_s 为系统总损耗因子， $0 < L_s < 1$ ， \tilde{G}_t 为发射天线的复电压增益， \tilde{G}_r 为接收天线的复电压增益， R 为目标斜距， $\lambda = \frac{c}{f_c}$ 为雷达工作波长， $f_c = \frac{2v}{\lambda}$ 为目标多普勒模型， v 为目标的径向速度， $b = |\tilde{b}| \exp(j\beta)$ 为目标散射参量， $|\tilde{b}| = \sigma$ 为目标的雷达横截面积， β 为目标的散射相位，在 $[0, 2\pi]$ 是均匀分布的。

3. 接收机噪声

可以将雷达接收机噪声看作是一个高斯过程的采样函数。带通噪声信号可表示为：

$$n(t) = \operatorname{Re} \left[\tilde{n}(t) e^{j2\pi f_c t} \right] \quad (6-3)$$

式中 $\tilde{n}(t) = n_d(t) - j n_q(t)$ ， $n_d(t)$ 和 $n_q(t)$ 是均值为零、方差为 σ_N^2 的独立高斯随机过程。噪声方差是由接收机噪声系数 N_F 和接收机带宽 B_R 计算的，即：

$$\sigma_N^2 = k T_o N_F B_R \quad (6-4)$$

式中 $k = 1.38 \times 10^{-23}$ 焦耳/度 为波尔兹曼常数， T_o 为接收机等效温度（290K）， B_R 为接收机带宽。

4. 杂波反射信号

一般情况下，认为旁瓣很低，可以忽略，这里只考虑主瓣杂波。杂波单元的反射信号可表示成复数形式：



(6-5)

$$C(t) = \text{Re}[\tilde{C}(t)e^{j2\pi f_c t}]$$

式中 $\tilde{C}(t) = \tilde{K} \tilde{g}_n \tilde{P}\left(t - \frac{2R_c}{c}\right)$, 其中, $\tilde{K} = \sqrt{\frac{P_t L_s}{(4\pi)^3}} \frac{\tilde{G}_t \tilde{G}_r \lambda}{R_c^2}$, \tilde{G}_t 为杂波单元中心方向上发射天线的复电压增益, \tilde{G}_r 为杂波单元中心方向上接收天线的复电压增益, R_c 为到杂波单元的斜距, \tilde{g}_n 为杂波复散射参量。

射天线的复电压增益, \tilde{G}_r 为杂波单元中心方向上接收天线的复电压增益, R_c 为到杂波单元的斜距, \tilde{g}_n 为杂波复散射参量。

通常 \tilde{g}_n 由一个固定分量加一个复高斯分量组成一个复散射参量, 对地杂波来说, 复高斯过程脉冲到脉冲的采样是相关的, 它是构造具有各种统计特性杂波的基础。

依照介绍的雷达系统框图建立其 Simulink 仿真框图如图 6-2 所示。由图可知系统分为五部分: 脉冲信号生成器、发射端射频前端、观测物体、接收端射频前端和信号处理单元。其中发射端射频前端和接收端射频前端涉及电路设计方面知识, 将在第 10 例中予以进一步介绍。而脉冲信号生成器分系统框图如图 6-3 所示, 观测物体内部框图如图 6-4 所示, 信号处理单元如图 6-5 所示。

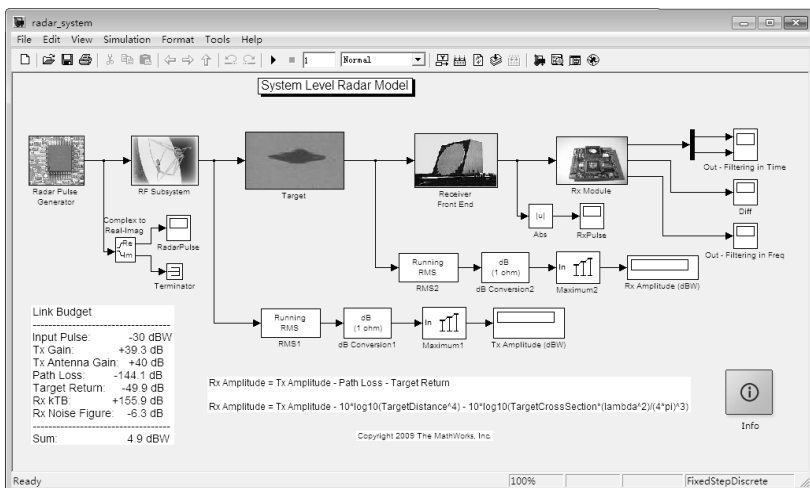


图 6-2 雷达系统 Simulink 仿真框图

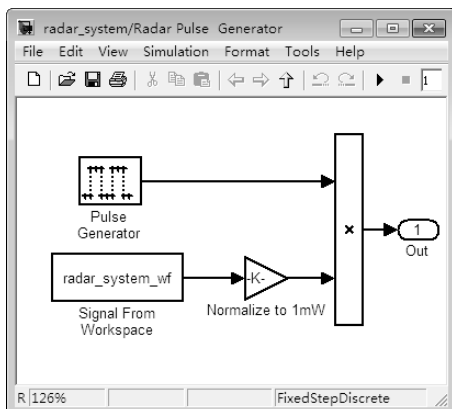


图 6-3 脉冲信号生成器框图

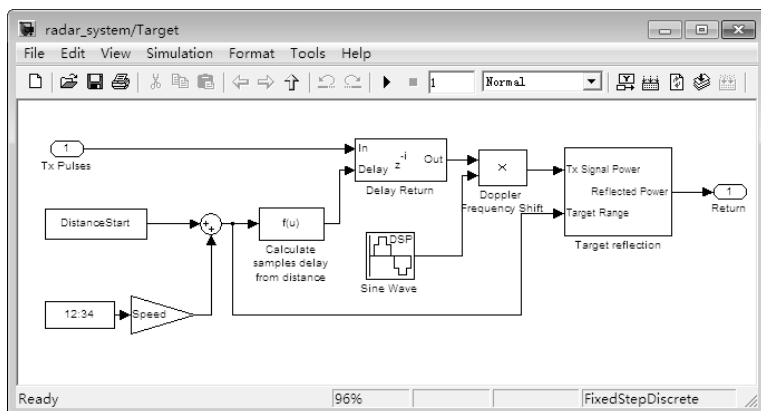


图 6-4 观测物体内部框图

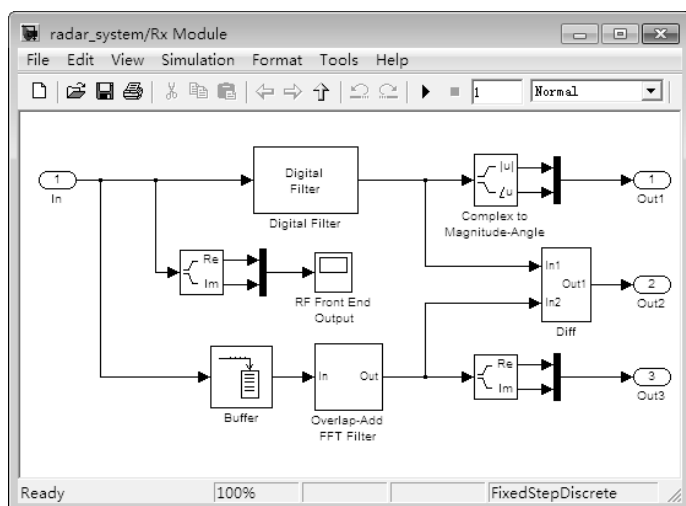


图 6-5 信号处理单元

仿真结果如图 6-6 所示。由图可知，雷达系统经过发射信号至观测物体，然后再返回雷达，通过处理能够辨识出信号，说明整个系统的有效性。

为进一步说明 Simulink 的功能，这里设计一个 S 函数来代替图 6-3 中“Pulse Generator”模块。这里给出 S 函数中的核心代码：

```
function sys=mdlOutputs(t,x,u)
x = t*10*50*128;
y = 50*128;
n = floor(x./y)
gg = x - n.*y;
if gg < 200
    sys = 1;
else
    sys = 0;
end
```

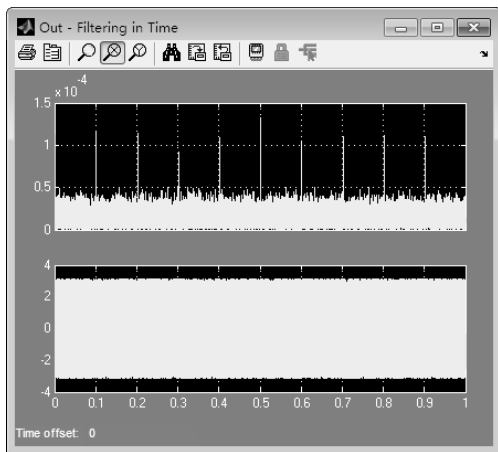


图 6-6 雷达反馈信号仿真结果

6.3 本例小结

本例以舰载雷达系统为对象，基于 MATLAB 通信工具箱模拟其信号产生、调频、反射、解调及信号处理过程，通过该例读者可以进一步了解和掌握基于 MATLAB 通信工具箱进行通信系统建模、仿真的特点及流程，对于雷达系统结构及原理也将有所了解。



第 7 例 机载GPS信号接收 及处理过程建模

20 世纪发射的 GPS 卫星使全球导航定位精度达到了一个新的高度，其在交通导航、军事战争等领域中已得到了充分的应用和验证。本例研究飞机上搭载 GPS 接收机接收和处理 GPS 信号的过程，此外，本例还介绍基于 MATLAB 文件操作的基本内容。

通过本例学习，需了解和掌握以下几点：

- 掌握基于 MATLAB 文件操作方法。
- 了解 GPS 信号接收原理。
- 掌握基于 MATLAB 对 GPS 接收过程进行建模仿真的方法。

7.1 基于 MATLAB 文件操作简介

文件操作是一种重要的输入/输出方式，即从数据文件读取数据或将结果写入数据文件。MATLAB 提供了一系列底层输入/输出函数，专门用于文件操作。

7.1.1 文件的打开与关闭

1. 打开文件

在读写文件之前，必须先用 `fopen` 函数打开或创建文件，并指定对该文件进行的操作方式。

`fopen` 函数的调用格式为：

`fid=fopen(文件名, '打开方式')`

说明：其中 `fid` 用于存储文件句柄值，如果返回的句柄值大于 0，则说明文件打开成功。文件名用字符串形式，表示待打开的数据文件。常见的打开方式如下。

- 'r'：只读方式打开文件（默认的方式），该文件必须已存在。
- 'r+'：读/写方式打开文件，打开后先读后写。该文件必须已存在。
- 'w'：打开后写入数据。该文件已存在则更新；不存在则创建。
- 'w+'：读/写方式打开文件。先读后写。该文件已存在则更新；不存在则创建。
- 'a'：在打开的文件末端添加数据。文件不存在则创建。
- 'a+'：打开文件后，先读入数据再添加数据。文件不存在则创建。

另外，在这些字符串后添加一个“t”，如‘rt’或‘wt+’，则将该文件以文本方式打开；如果添加的是“b”，则以二进制格式打开，这也是 fopen 函数默认的打开方式。

2. 关闭文件

文件在进行完读、写等操作后，应及时关闭，以免数据丢失。关闭文件用 fclose 函数，调用格式为：

```
sta=fclose(fid)
```

说明：该函数关闭 fid 所表示的文件。sta 表示关闭文件操作的返回代码，若关闭成功，返回 0，否则返回-1。如果要关闭所有已打开的文件，用 fclose('all')。

7.1.2 二进制文件的读/写操作

1. 写二进制文件

fwrite 函数按照指定的数据精度将矩阵中的元素写入到文件中。其调用格式为：

```
COUNT=fwrite ( fid , A , precision )
```

说明：其中 COUNT 返回所写的的数据元素个数（可默认），fid 为文件句柄，A 用来存放写入文件的数据，precision 代表数据精度，常用的数据精度有：char、uchar、int、long、float、double 等。默认数据精度为 uchar，即无符号字符格式。

例 7.1 将一个二进制矩阵存入磁盘文件中。

```
>> a=[1 2 3 4 5 6 7 8 9];
>> fid=fopen('d:\test.bin','wb')    %以二进制数据写入方式打开文件
fid =
    3                                %其值大于 0，表示打开成功
>> fwrite(fid,a,'double')
ans =
         9                            %表示写入了 9 个数据
>> fclose(fid)
ans =
         0                            %表示关闭成功
```

2. 读二进制文件

fread 函数可以读取二进制文件的数据，并将数据存入矩阵。其调用格式为：

```
[A , COUNT]=fread(fid , size , precision)
```

说明：其中 A 是用于存放读取数据的矩阵、COUNT 是返回所读取的数据元素个数、fid 为文件句柄、size 为可选项，若不选用则读取整个文件内容；若选用则它的值可以是下列值：N（读取 N 个元素到一个列向量）、inf（读取整个文件）、[M, N]（读数据到 M×N 的矩阵中，数据按列存放）。precision 用于控制所写数据的精度，其形式与 fwrite 函数相同。



7.1.3 文本文件的读/写操作

1. 读文本文件

fscanf 函数可以读取文本文件的内容，并按指定格式存入矩阵。其调用格式为：

`[A, COUNT]=fscanf(fid, format, size)`

说明：其中 A 用来存放读取的数据，COUNT 返回所读取的数据元素个数，fid 为文件句柄，format 用来控制读取的数据格式，由 % 加上格式符组成，常见的格式符有：d（整型）、f（浮点型）、s（字符串型）、c（字符型）等，在 % 与格式符之间还可以插入附加格式说明符，如数据宽度说明等。size 为可选项，决定矩阵 A 中数据的排列形式，它可以取下列值：N（读取 N 个元素到一个列向量）、inf（读取整个文件）、[M,N]（读数据到 M×N 的矩阵中，数据按列存放）。

2. 写文本文件

fprintf 函数可以将数据按指定格式写入到文本文件中。其调用格式为：

`fprintf(fid, format, A)`

说明：fid 为文件句柄，指定要写入数据的文件，format 是用来控制所写数据格式的格式符，与 fscanf 函数相同，A 是用来存放数据的矩阵。

例 7.2 创建一个字符矩阵并存入磁盘，再读出赋值给另一个矩阵。

```
>> a='string';
>> fid=fopen('d:\char1.txt','w');
>> fprintf(fid,'%s',a);
>> fclose(fid);
>> fid1=fopen('d:\char1.txt','rt');
>> fid1=fopen('d:\char1.txt','rt');
>> b=fscanf(fid1,'%s')
    b =
      string
```

7.1.4 MATLAB 读 txt 文件

MATLAB 读取 txt 文件一般用 fscanf 函数，其使用如下：

```
fid=fopen('fx.txt','r');
%得到文件号
[f,count]=fscanf(fid,'%f %f',[12,90]);
%把文件号 1 的数据读到 f 中。其中 f 是 [12 90] 的矩阵
%这里 '%f %f' 表示读取数据的形式，它是按原始数据类型读出的
fclose(fid);
%关闭文件
```

另外有的 txt 文件还可以用 load 来打开，其语句为：

```
f=load('fx.txt')
```



7.2 机载 GPS 信号接收及处理建模

为了跟踪和解码 GPS 信号，首先要捕获到 GPS 信号。将捕获到的 GPS 信号的必要参数立刻传递给跟踪过程，再通过跟踪过程便可得到卫星的导航电文。GPS 卫星处于高速运动中，因此，其频率会产生多普勒频移。为覆盖高速卫星预期中的所有多普勒频移范围，捕获方法覆盖的频率范围必须在 $\pm 10\text{kHz}$ 之内。一旦捕获到 GPS 信号，立刻去测量两个重要参数，C/A 码的起始点和载波频率（因多普勒频移而变化）。接收机接收到的一系列数据往往包含多个卫星信号，每个信号具有不同的 C/A 码的不同起始点和不同的多普勒频率。针对某个特定的卫星信号，捕获过程就是要找到 C/A 码的起始点，并利用找到的起始点展开 C/A 码频谱，一旦复现了 C/A 码的频谱，输出信号将变成连续波，于是便得到其载波频率。即捕获过程就是要获得输入信号的 C/A 码的起始点和载波频率，然后传递给跟踪过程。

捕获与跟踪过程所用到的数据都是从原始的卫星信号经过下变频器（即与中频混合）之后收集到的，其中频为 21.25MHz ，采样频率为 5MHz ，信号的中心频率为 1.25MHz 。

这里对 GPS 卫星信号的捕获采用循环捕获的方法。在实际捕获中，输入电文不是连续到达接收机的，因此循环相关操作适合一组或一批电文。输入电文经过 5MHz 的 ADC（模拟/数字转换器）存储在存储器中，只有 1ms 的输入电文用来寻找 C/A 码的起始点，其搜索频率分辨率即步进频率是 1MHz 。

捕获输入电文，需经过以下几个步骤。

(1) 对 1ms 的输入电文 $x(n)$ 进行快速傅里叶变换 (FFT)，将输入转换到频域，值为 $X(k)$ ，这里， $n = k = 0, 1, 2, \dots, 4999$ 。

(2) 取 $X(k)$ 的复共轭，值为 $X(k)^*$ 。

(3) 利用式 (7-1) 产生 21 个本地码 $l_{si}(n) (i=1, 2, \dots, 21)$ ，本地码包含卫星 s 的 C/A 码和一个复射频信号的乘积，然后用 5MHz 采用本地码，本地码的频率 f_i 相距 1MHz 。

$$l_{si} = C_s e^{j2\pi f_i t} \quad (7-1)$$

式中， s 表示卫星的编号； C_s 表示编号为 s 卫星的 C/A 码； $f_i = 1250 - 10, 1250 - 9, \dots, 1250 + 10\text{kHz}$ 。

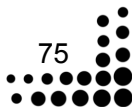
(4) 对 $l_{si}(n)$ 进行 FFT，变换到频域中，值为 $L_{si}(k)$ 。

(5) 将 $X(k)^*$ 与 $L_{si}(k)$ 点对点相乘，结果为 $R_{si}(k)$ 。

(6) 对 $R_{si}(k)$ 进行反傅里叶变换，变换到时域中的值 $r_{si}(n)$ ，得到其绝对值 $|r_{si}(n)|$ 。

(7) 在输入电文 200ns 的时间分辨率和载波频率为 1kHz 分辨率的条件下， $|r_{si}(n)|$ 最大值中的第 n 位和第 i 个载波频率给出了 C/A 码的初始点。

从 1ms 电文中得到的频率分辨率大约是 1kHz ，对跟踪环来说，这个值太粗糙了。适合跟踪过程的频率必须在几十赫兹之内。通常，跟踪环的频宽是几赫兹，用 DFT (或 FFT) 计算精细频率 (Fine Frequency) 是不可取的。如果采用长电文方式来提高频率精度将会带来计算复杂费时的缺点。





对于固定电文长度提高精细频率分辨率的方法是通过相位关系。一旦输入信号剥离了 C/A 码,输入就将变成连续波形。如果在 m 时刻,1ms 电文中最高频率分量是 $X_m(k)$ (k 表示输入信号的频率分量),则可由 DFT 输出得到输入信号的初始相位 $\theta_m(k)$:

$$\theta_m(k) = \tan^{-1} \left(\frac{\text{Im}(X_m(k))}{\text{Re}(X_m(k))} \right) \quad (7-2)$$

Im 和 Re 分别表示虚部和实部。假设在 m 时刻之后很短时间的 n 时刻,1ms 电文的 DFT 分量 $X_n(k)$ 也是最强分量,因为输入分量在很短时间内不会迅速变化。 n 时刻输入信号的初始相位角和频率分量 k 为:

$$\theta_n(k) = \tan^{-1} \left(\frac{\text{Im}(X_n(k))}{\text{Re}(X_n(k))} \right) \quad (7-3)$$

这两个相位角可用来计算精频:

$$f = \frac{\theta_n(k) - \theta_m(k)}{2\pi(n-m)} \quad (7-4)$$

这个方程式给出了一个比 DFT 得到的要精确得多的频率分辨率。为了保持其值的唯一性, $\theta_n(k) - \theta_m(k)$ 的相位差必须小于 2π 。如果相位差是最大值 2π ,带宽就是 $1/(n-m)$ 。这里的 $(n-m)$ 是两组连续电文之间的延时。

具体来说,在某一确定卫星中需要经过下面几个步骤来找出它的 C/A 码起始点和载波频率。

(1) 对 1ms 的输入执行循环相关操作,某个确定的 C/A 码起始点可以从这些循环相关里找到,载波频率可以以 1kHz 的分辨率得到。

(2) 找到最高频率分量 $X(k)$,在同一毫秒数据内的两个分量,一个比 $X(k)$ 中的 k 值低 400kHz,另一个比 k 值高 400kHz,对这两个分量执行 DFT 操作。三个输出 $[X(k-1), X(k), X(k+1)]$ 中的最高输出将被指定为新的 $X(k)$,并用这个新的 $X(k)$ 作为 DFT 分量来求解精频。

(3) 从 C/A 码起始点处开始选择连续几个毫秒的数据,随意选择为 5ms,将这些数据与 5 组连续 C/A 码相乘,结果肯定是一个 5ms 长的连续信号。但是在任意 1ms 数据中,都可能含有一个 π 相位偏移。

(4) 从所有输入数据中找出 $X_n(k)$,其中 $n=1,2,3,4,5$,然后找出相位角。相位角定义为: $\Delta\theta = \theta_{n+1} - \theta_n$ 。

(5) 角度差的绝对值必须小于门限值 ($2.3\pi/5$),如果不能实现,就要从 $\Delta\theta$ 上加上或减去 π ,得到的值还要与门限值 $2.3\pi/5$ 进行比较,来决定是否要再次加减 2π 。经过这些调整后,最终的角度值就是期望值。

(6) 利用公式来计算精频。由于有 5ms 的数据,将得到 4 个精频值。为了提高精确度,将这 4 个精频的平均值作为要求解的值。

下面附上随机编码和获取导航信息的两段程序源代码。

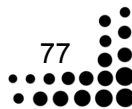
1) 随机编码过程仿真



```
close all; clear; clc;
t = 0:5:1000;
x1 = rand(1,length(t));
x1 = round(x1);
index1 = find(x1 == 0);
x1(index1) = -ones(1,length(index1));
x2 = rand(1,length(t));
x2 = round(x2);
index1 = find(x2 == 0);
x2(index1) = -ones(1,length(index1));
x3 = rand(1,length(t));
x3 = round(x3);
index1 = find(x3 == 0);
x3(index1) = -ones(1,length(index1));
y1 = zeros(1,1); y2 = zeros(1,1); y3 = zeros(1,1);
z1 = zeros(1,1); z2 = zeros(1,1); z3 = zeros(1,1);

% 将生成的伪随机数值存放到对应的矩阵中
y1(1) = x1(1); y2(1) = x2(1); y3(1) = x3(1);
t1(1) = t(1); t2(1) = t(1); t3(1) = t(1);
k = 2; m = 2; n = 2;

for i = 2:length(t)
    if (x1(i) == x1(i-1))
        y1(k) = x1(i);
        t1(k) = t(i);
        k = k+1;
    else
        y1(k) = x1(i-1);
        y1(k+1) = x1(i);
        t1(k) = x1(i);
        t1(k+1) = t(i)+0.01; % 如果该时刻阶跃变化，则右移 0.01 来记录
        k = k+2;
    end
    if (x2(i) == x2(i-1))
        y2(m) = x2(i);
        t2(m) = t(i);
        m = m+1;
    else
        y2(m) = x2(i-1);
        y2(m+1) = x2(i);
        t2(m) = t(i);
        t2(m+1) = t(i)+0.01;
        m = m+2;
    end
    if (x3(i) == x3(i-1))
        y3(n) = x3(i);
        t3(n) = t(i);
        n = n+1;
    end
end
```





```
else
    y3(n) = x3(i-1);
    y3(n+1) = x3(i);
    t3(n) = t(i);
    t3(n+1) = t(i)+0.01;
    n = n+2;
end
end

subplot(3,1,1); plot(t1,y1,'y'); grid on; axis([-1,1001,-1.5,1.5]);
subplot(3,1,2); plot(t2,y2,'y'); grid on; axis([-1,1001,-1.5,1.5]);
subplot(3,1,3); plot(t3,y3,'y'); grid on; axis([-1,1001,-1.5,1.5]);
whitebg('black');
simin1 = [t1; y1]'; simin2 = [t2; y2]'; simin3 = [t3; y3]';
```

2) 获取导航信息的仿真

在这个仿真中，卫星信号的数据是基于前面章节中得到的卫星数据，存放在 GPSsignal.mat 文件中。在运行程序之前，需将 GPSsignal.mat 中的数据传送到 Workspace 中。该文件在下一例中给出。

```
clear ; clc; close all;
load GPSsignal.mat

SvNum = 12;
% 调用上一例中的 fGenerateCAcode3.m 文件，获得卫星编号为 12 的 C/A 码
Temp = fGenerateCAcode3(SvNum);
index1 = find(Temp == 0); % 找出 C/A 码中的低电平，形成列向量存放在 index1 中
Temp(index1) = -ones(1,length(index1));
SinWave = sin([0:2*pi/2:2*pi*7/8]);
SinWave = single(SinWave);
GpsMatch = zeros(1,1);
SinWave = [SinWave SinWave SinWave SinWave SinWave];
for i = 1:length(Temp)
    GpsMatch = [GpsMatch Temp(1,i)*SinWave];
end
GpsMatch = GpsMatch(2:length(GpsMatch));
n = length(GpsMatch);
m = 50000;
for i = 1:m
    Res(i) = GpsMatch*GPSsignal(1,i:i+n-1)';
end
plot(1:m,Res);

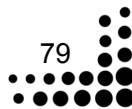
Res2 = abs(Res);
for i = 1:100
    [C I] = max(Res2);
    Res2(1,I) = 0;
    Index(1,i) = I;
end
```



```
% 下面要找到 GPSSignal 中的导航电文
w = length(GpsMatch);
m = 1;
for i = Index(1,1):w:(length(GPSSignal) - w + 1)
    NavigationBit(m) = (GPSSignal(i:i+w-1)*GpsMatch')/w;
    m = m+1;
end
NavigationCode = zeros(1,1);
NavigationCode(1,1) = NavigationBit(1,1);
m = 2; Count = 0;

for i = 2:length(NavigationBit)
    if(NavigationBit(1,i)~= NavigationBit(1,i-1))
        NavigationCode(1,m)= NavigationBit(1,i);
        m = m+1;
        Count = 0;
    else
        Count = Count+1;
        if(Count>=5)
            NavigationCode(1,m)= NavigationBit(1,i);
            m = m+1;
            Count = 0;
        end
    end
end

% 将得到的导航码 NavigationCode 转换成数字
NavigationCode = NavigationCode./abs(NavigationCode);
index1 = find(NavigationCode<0);
NavigationCode(index1) = zeros(1,length(index1));
Table1 = [0 0 0 0; 0 0 0 1; 0 0 1 0; 0 0 1 1; 0 1 0 0; 0 1 0 1; 0 1 1 0;
0 1 1 1; 1 0 0 0; 1 0 0 1;...
1 0 1 0; 1 0 1 1; 1 1 0 0; 1 1 0 1; 1 1 1 0; 1 1 1 1;];
Result = '';
for i = 1:4:length(NavigationCode)
    TT = NavigationCode(i:i+3);
    MatchTable = Table(:,1:4)-[TT; TT; TT; TT; TT; TT; TT; TT; TT; TT;
TT; TT; TT; TT; TT];
    MatchTable = sum(abs(MatchTable)');
    TempChar = '0';
    BestMatch = find(MatchTable == min(MatchTable));
    switch(BestMatch)
        case 1 TempChar = '0';
        case 2 TempChar = '1';
        case 3 TempChar = '2';
        case 4 TempChar = '3';
        case 5 TempChar = '4';
        case 6 TempChar = '5';
        case 7 TempChar = '6';
    end
end
```





```
case 8 TempChar = '7';  
case 9 TempChar = '8';  
case 10 TempChar = '9';  
case 11 TempChar = 'a';  
case 12 TempChar = 'b';  
case 13 TempChar = 'c';  
case 14 TempChar = 'd';  
case 15 TempChar = 'e';  
case 16 TempChar = 'f';  
end  
Result = stract(Result, TempChar);  
end  
Num1 = hex2num(Result(1:16));  
Num2 = hex2num(Result(17:32));  
Num3 = hex2num(Result(33:48));
```

7.3 本例小结

本例以机载 GPS 接收机为对象，基于 MATLAB 通信工具箱模拟其捕获和跟踪 GPS 信号的过程，通过该例读者可以了解和掌握基于 MATLAB 通信工具箱进行信号捕获跟踪建模、仿真的特点及流程，对于 GPS 信号捕获接受方法及过程也将有所了解。



第 8 例 GPS 卫星发射信号模拟

上一例对 GPS 信号接收及捕获予以了介绍，本例介绍 GPS 信号生成过程，此外对串口操作基本知识也予以了介绍。通过本例学习，需了解和掌握以下几点：

- 了解和掌握串口操作命令及过程。
- 了解 GPS 中 C/A 码生成的基本原理。
- 了解 GPS 中导航电文生成的基本原理。
- 掌握基于 MATLAB 生成 C/A 码和导航电文的方法。

8.1 MATLAB 串口操作简介

MATLAB 串口通信技术属于 MATLAB 仪器控制工具箱一部分。相比于 VB 或 C/C++ 语言，通过 MATLAB 进行串口控制更易于上手，编程方便，最大的优点是结合 MATLAB 自身强大的数据处理分析能力，再配合 Simulink 模块的支持，便可以用同一软件实现产品软硬件的联合测试，数据算法等研究任务。同时 MATLAB 还自带采集工具箱，GUI 界面等模块，使其成为一个功能全面的软件，从而提高了工作效率，节约了成本。

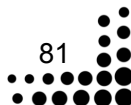
1. 创建设备对象

MATLAB 将设备视为对象，分为界面对象和驱动对象。在使用对象前需要先连接对象。

- 连接到对象使用函数 `fopen`。
- 断开对象 `fclose`。
- 连接到驱动对象使用函数 `connect`。
- 断开使用 `disconnect`。
- 检查对象状态使用函数 `obj.status`。
- 获得对象属性名称和属性值使用命令 `get`，例如：`get(g,'OutputBufferSize')`。
- 设置对象属性名称的属性值使用命令 `set`，例如：`set (g,'EOSMode','read')`。
- 将设备从内存中删除 `delete(obj)`。
- 将设备从工作空间中删除 `clear(obj)`。
- 恢复设备 `instrfind`。

2. 设备写数据

- `binblockwrite` 向设备写入 `binblock` 数据。





- fprintf 向设备写入文本文件。
- fwrite 向设备写入二进制文件。
- stopasync 停止异步读/写操作。

与写操作相关的属性有：

- BytesToOutput 指出当前输出缓冲区数据位数。
- OutputBufferSize 指定输出缓冲区大小。
- Timeout 读/写等待时间。
- TransferStatus 指出读/写是否正在进行。
- ValuesSent 指出写入设备值的总数。

3. 同步和异步操作

- 默认情况下全部为同步操作，如果想实现异步操作，则要在 fprintf 和 fwrite 命令后加入后缀“ async”。例如：fwrite(g,[cmd data], 'async')。
- TransferStatus 函数可以检查异步通信的状态。例如：g.TransferStatus。
- BytesToOutput 函数可以用来检查在缓冲区等待传送的数据个数。例如：g.BytesToOutput。

4. 从设备读数据

- binblockread 从设备读取 binblock 数据。
- fgetl 从设备读取一行文本并抛弃结尾。
- fgets 从设备读取一行文本包含结尾。
- fread 从设备读取二进制数据。
- fscanf 从设备读取数据，并格式化为文本。
- readasync 从设备异步读取数据。
- scanstr 从设备读取数据，格式化为文本并分析。
- stopasync 停止异步读/写操作。

与读数据有关的属性：

- BytesAvailble 指出输入缓冲区的有效数据个数。
- InputBufferSize 指定输入缓冲区大小。
- ReadAsyncMode 指定异步读取是连续的还是手动的。
- Timeout 指定读/写操作的等待时间。
- TransferStatus 指出异步读/写操作是否正在进行。
- ValuesReceived 指出从设备读取值的总数。

5. 同步和异步读取操作

函数 fgetl, fgets, fscanf 和 fread 函数的操作都是同步的。如果要执行异步读取操作，要使用 readasync 函数。readasync 从设备异步读取数据并保存在输入缓冲区中。把数据从

输入缓冲区读入 MATLAB 变量，就要使用同步读取函数。

6. 通过串口控制设备

- 创建串口对象，这里使用函数 serial。例如：`s = serial('COM1')`。
- 显示其属性值使用 get。例如：`get(s,{'Name','Port','Type'})`。

相关串口通信属性有：

- BaudRate 波特率。
- DataBits 指定传送数据的位数。
- Parity 指定奇偶校验。
- StopBits 指定停止位位数。
- Terminator 指定结尾字符。

例如：`get(s,{'BaudRate','DataBits','Parity','StopBits','Terminator'})`。

8.2 GPS 的 C/A 码及导航电文建模

GPS 发射的测距码信号包括 C/A 码和 P 码，它们都是二进制伪随机噪声序列，具有特殊的统计特性。

1. 码的基本概念

码是一种表达信息的二进制数及其组合，是一组二进制的数码序列。例如，对 0, 1, 2, 3 取两位二进制数的不同组合表示为：00, 01, 10, 11。这些二进制数的组合形式称之为码。其中每一位二进制数称为 1 个码元或 1 比特 (bit)；每个码均含有两个二进制数，即两个码元或两个比特。比特是码的度量单位，也是信息量的度量单位。如果将各种信息，例如声音、图像以及文字等，按照某种预定的规则表示为二进制数的组合形式，则这一过程就称为编码，也就是信息的数字化。

在二进制的数字化信息传输中，每秒所传输的比特数称为数码率，用以表示数字化信息的传输速度，其单位为 bit/s (简称为 b/s)。码可以看作是以 0 和 1 为幅度的时间函数，用 $u(t)$ 表示。因此，一组码序列 $u(t)$ ，对应某个时刻 t ，码元是 0 或者 1 完全是随机的，但其出现的概率均为 1/2。这种码元幅值是完全无规律的码序列，称为随机噪声码序列。它是一种非周期序列，无法复制。但是，随机噪声序列具有良好的自相关性，GPS 测距码就是利用其自身良好的自相关性才获得成功的。

这里，自相关性是指两个结构相同的码序列的相关程度，它由自相关函数描述。为了说明这一问题，可将随机噪声码序列平移 k 个码元，获得具有相同结构的新码序列 $u(t)$ 。比较这两个码序列，假定它们对应的码元中，码值相同的个数为 S_u ，而码元相异的个数为 D_u ，那么两者之差和两者之和的比值定义为随机噪声码序列的自相关函数，用符号 $R(t)$ 表示：



$$R(t) = \frac{S_u - D_u}{S_u + D_u} \quad (8-1)$$

在实际应用中, 可通过自相关函数的取值判断两个随机噪声码序列的相关性。当平移的码元个数为 0 时, 两个结构相同的码序列对应码元完全相同, 这时 $D_u = 0$, 而自相关函数 $R(t) = 1$; 相反, 当 $k \neq 0$ 时, 且假定码序列中码元总数特别大, 那么由于码序列的随机性, 将有 $S_u \approx D_u$, 这时自相关函数 $R(t) \approx 0$ 。因此, 根据自相关函数 $R(t)$ 的取值, 可确定两个随机噪声码序列是否已经“相关”, 或者两个码序列的对应码元是否已经完全“对齐”。

假设 GPS 卫星发射的一个随机序列 $u(t)$, 而 GPS 信号接收机在收到信号的同时复制出结构与 $u(t)$ 完全相同的随机序列 $\tilde{u}(t)$, 由于信号传播延迟的影响, 被接收的随机序列 $u(t)$ 与 $\tilde{u}(t)$ 之间产生了平移, 即对应码元已错开, 因而 $R(t) \approx 0$ 。若通过一个时间延迟器来调整, 测出卫星信号到达用户接收机的准确传播时间, 再乘以光速便可确定卫星至观测站的距离。所以, 随机噪声序列码良好的自相关特性为 GPS 测距奠定了基础。

2. 随机噪声序列码的产生

伪随机噪声码简称 PRN 码, 是一个具有一定周期的取值 0 和 1 的离散符号串。它不仅具有高斯噪声所有的相关特性, 而且具有某种确定的编码规则。它是周期性的、可人工复制的码序列。GPS 信号中使用了伪随机噪声编码技术, 识别和分离各颗卫星信号, 并提供无模糊度的测距数据。GPS 信号中的 C/A 码和 P 码, 都是由最长线性移位寄存器序列 (简称 m 序列) 产生的伪随机测距码。

随机码由多级反馈移位寄存器产生。这种寄存器由一组连接在一起的存储单元组成, 每个存储单元只有“0”或“1”两种状态, 并接收时脉冲和置“1”脉冲的驱动及控制。一般来说, 一个 r 级移位寄存器所产生的 m 序列, 在一个周期内其码元的最大个数为:

$$N_u = 2^r - 1 \quad (8-2)$$

与此对应, 这时 m 序列的最大周期为:

$$T_u = (2^r - 1)t_u \quad (8-3)$$

由于移位寄存器不容许出现全“0”状态, 因此 $2^r - 1$ 码元中, “1”的个数总比“0”的个数多一个。这样, 当两个周期相同的 m 序列其对应码元完全对齐时, 自相关系数 $R(t) = 1$, 而在其他情况有:

$$R(t) = -\frac{1}{N_u} = -\frac{1}{2^r - 1} \quad (8-4)$$

当 r 够大时, 有 $R(t) \approx 0$ 。所以, 伪随机噪声码和随机噪声码一样, 具有良好的自相关性, 而且是一种结构确定、可以复制的周期性序列。GPS 信号接收机就是利用这一特征使所接收的伪随机噪声码和机内产生的伪随机噪声码达到对齐同步, 进而捕获和识别来自不同的 GPS 卫星的伪随机噪声序列的。由于受 GPS 卫星至用户 GPS 接收机的路径信号传播延迟的影响, 被接收的伪随机码和复制的伪随机码之间产生了平移; 如果通过一个时间延迟器来对复制的伪随机进行移动, 使两者的相关函数值为 1, 则可以从时间延



迟器中测出对齐码元所用的时间,从而可以较准确地确定由卫星到接收机的距离。由此可见,伪随机序列的良好的自相关特性,对于利用 GPS 卫星的测距码进行精密测距具有重要的意义。

3. GPS 测距码信号

C/A 码 (Coarse Acquisition Code) 是用于粗测距和捕获 GPS 卫星信号的伪随机码。它是由 10 级反馈移位寄存器组合产生的。

两个移位寄存器于每星期日子夜零时,在置“1”脉冲作用下处于全“1”状态,同时在频率为 $f_1 = f_0/10 = 1.023\text{MHz}$ 时钟驱动下,两个移位寄存器分别产生码长为 $N = 2^{10} - 1 = 1023$ 、周期为 1ms 的两个 m 序列 $G_1(t)$ 和 $G_2(t)$ 。这时 $G_2(t)$ 序列的输出不是在该移位寄存器的最后一个存储单元,而是选择其中两个存储单元进行二进制相加后的输出,由此得到一个与 $G_2(t)$ 平移等价的 m 序列 G_{21} 。再将其与 $G_1(t)$ 进行模二相加,将可能产生 1023 种不同结构的 C/A 码。C/A 码不是简单的 m 序列,而是由两个具有相同码长及数码率,但结构不同的 m 序列相乘所得到的组合码,称为戈尔德 (Gold) 码。

$$C/A(t) = G_1(t) \cdot G_2(t + it_0) \quad (8-5)$$

采用不同的 it_0 值,可能产生 1023 个 $G_2(t)$,再加上 $G_1(t)$ 和 $G_2(t)$ 本身,共可能产生 1025 个不同的 C/A 码供选用。这些 C/A 码具有相同的码长 相同的码元宽 $t_u = 1/f_1 = 0.98\mu\text{s}$ 和相同的周期 $T_u = Nt_u = 1\text{ms}$ 。

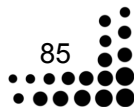
从这些 $G(t)$ 码中选择 32 个码以 PRN1、PRN2, ..., PRN32 命名各种 GPS 卫星。由于 C/A 码很短,只有 1023 比特,易于捕获。在 GPS 定位中,为了捕获 C/A 码,以测定卫星信号的传播延时,通常需要对 C/A 码逐个进行搜索。若以 50 个码元每秒的速度搜索,对于只有 1023 个码元的 C/A 码,搜索时间只需要 20.5s。通过 C/A 码捕获卫星后,即可获得导航电文,通过导航电文提供的信息,便可以很容易地捕获 GPS 的 P 码。所以除了作为粗测码外,还可作为 GPS 卫星信号 P 码的捕获码。

C/A 码的码元宽度较大。假设两个序列的码元对齐误差为码宽的 1/10~1/100,则此时相应的测距误差为 29.3~2.93m。随着相关技术发展,使得测距分辨率大大提高。一般最简单的导航接收机的伪距测量分辨率可达 0.1m。不同的 GPS 卫星所使用的 C/A 码指标相同,但编码规则不同,这样既便于复制又便于区分。

4. GPS 卫星的导航电文

导航电文的基本单位是一个主帧,每个主帧长度为 1500bit,由 5 个子主帧组成。每个子主帧分别含有 10 个字,每个字含 30bit 电文,故每一个子帧含有 10 个字,每个字含有 30bit 电文,故每一个子帧共含 300bit 电文。电文的传输速率是 50b/s,所以播发一帧电文需要 30s,而一子帧电文的持续播发时间为 6s。

为了记载多达 25 颗 GPS 卫星的星历,规定 4、5 子帧各有 25 个页面。子帧 1、2、3 与子帧 4、5 的每一页均构成一帧电文。每 25 帧导航电文组成一个主帧。在每一帧电文中,1、2、3 子帧的内容每 30s 重复一次,每小时更新一次,而子帧 4、5 的内容仅在为卫星注入新的导航数据后才得以更新。





GPS 卫星的导航电文是卫星以二进制的形式发送给用户的导航定位数据，故又称为数据码或 D 码，是用户用来定位和导航的数据基础。它主要包括：卫星星历、时钟改正、电离层时延改正、工作状态信息及 C/A 码转换到捕获 P 码的信息、全部卫星的概略星历。

第 1、2、3 子帧播放该卫星的广播星历及卫星时钟的修正参数，每 30s 重复一次，内容每小时更新一次。第 4、5 子帧播放所有空中 GPS 卫星的历书（卫星的概略坐标），完整的历书占 25 帧，由于播放速度是 50b/s，所以全部信息需要 12.5min 才能够传送完。其内容仅在卫星注入新的导航数据后才得以更新。

附程序：

```
% 产生 C/A 码的方法一
k1=2;k2=6;delay=5;
Reg=-ones(1,10);
for j=1:1023;
    MLS(j)=Reg(10);
    modulo= Reg(2)* Reg(3)* Reg(6)* Reg(8)* Reg(9)* Reg(10);
    Reg(2:10)=Reg(1:9);
    Reg(1)=modulo;
    g2(j)=Reg(k1)*Reg(k2);
end
% 将 G2 和 MLS 进行延迟检测
if MLS==g2([delay:1023 1:delay-1])
    disp('OK')
else
    disp('not match')
end
% 在 G2 序列中找出-1 并转换为 0，找出 1 并转换为 1
ind1=find(g2==-1);
ind2=find(g2==1);
g2(ind1)=ones(1,length(ind1));
g2(ind2)=ones(1,length(ind2));
temp=g2(1:120);
x(1)=0; Show(1)=temp(1);
% 下面的循环是为了将结果显示成方波形式
for l=2:length(temp)
    if(temp(i)==temp(i-1))
        x(P)=i-1;
        Show(P)=temp(i-1);
        x(P+1)=i-1+0.1;
        Show(P+1)=temp(i);
        P=P+2;
    else
        Show(P)=temp(i);
        x(P)=i;
        P=P+1;
    end
end
% 画出仿真结果图
```



```
plot(x,Show);
axis([0 length(x)-60 -0.1 1.1]);
grid;
```

5. 卫星数据 GPSSignal

假设每个导航数据包包含 32 位，每个导航数据包包含 20 个 C/A 码组（1023 位），每组 C/A 码的周期为 1ms，载频为 154F（L1 载波）和 120F（L2 载波）两种，则每位含 1200 或 1540 个载波。当每个载波按 8 个采样点计算时，数据量将会太大，导致仿真速度太慢，从而无法仿真。若假设每个导航数据包包含 32 位，而每个导航数据包只包含 5 个 C/A 码组，每个 C/A 码组仅包含 10 个载波，当每个载波按 8 个采样点计算时，每个导航数据包仿真数据为：

$$32 \times 5 \times 1023 \times 10 \times 8 = 13\text{M}$$

可见数据量将少很多。

```
%%
x1 = 10.23; y1 = 6.24; z1 = 17.8; % 假设卫星的位置参数
% Code1 由子函数 fGenerateNavigationData 实现
Code1 = fGenerateNavigationData(x1,y1,z1);
index1 = find(Code1 == 0);
Code1(index1) = -ones(1,length(index1));
SvNum = 12; % 设定卫星编号为 12
Code2 = zeros(1,1); % 定义 Code2 的初值为 0
% 将编号为 SvNum 的卫星通过调用子函数 fGenerateCAcode3 生成 C/A 码
Temp = fGenerateCAcode3(SvNum);
% 将 Temp 中的 0 找出并转换为 -1
index1 = find(Temp == 0);
Temp(index1) = -ones(1,length(index1));
Temp = [Temp Temp Temp Temp Temp];
% 生成 Code2
for i = 1:length(Code1)
Code2 = [Code2 Code1(1,i)*Temp];
end
Code2 = Code2(2:length(Code2));
% 每位数据通过正弦波来调制
SinWave = sin([0:2*pi/8:2*pi*7/8]);
SinWave = single(SinWave);
GPSSignal = zeros(1,1);
SinWave = [SinWave SinWave SinWave SinWave SinWave];
for i = 1:length(Code2)
GPSSignal = [GPSSignal Code2(1,i)*SinWave];
end
GPSSignal = GPSSignal(2:length(GPSSignal));
% 将卫星数据 GPSSignal 存入 GPSSignal.mat 文件中
save GPSSignal.mat GPSSignal
```

```
% 子函数 fGenerateCAcode3
function y = fGenerateCAcode3(svnum)
```





```
g2s = [5;6;7;8;17;18;139;140;141;251;252;254;255;256;257;258;469; 470;
472;473;474;509;512;513;514;515;516;859;860;861;862];
g2shift = g2s(svnum,1);
reg = -1*ones(1,10);
for i = 1:100
    g(i) = reg(10);
    slavel = reg(3)*reg(10);
    reg(1,2:10) = reg(1:1:9);
    reg(1) = slavel;
end
reg = -1*ones(1,10);
Table1 = [ 2 6; 4 8; 5 9; 1 9; 2 10; 1 8; 2 9; 3 10; 2 3; 3 4; 5 6; 6 7;
7 8; 8 9; 9 10; 1 4; 2 5; 3 6; 4 7; 5 8; 6 9; 1 3; 4 6;];
for i = 1:100
    save2 = reg(2)*reg(3)*reg(6)*reg(8)*reg(9)*reg(10);
    g2(i) = reg(Table1(svnum,1))*reg(Table1(svnum,2));
    reg(1,2:10) = reg(1:1:9);
    reg(1) = save2;
end
% 将 G1 和 G2 卷积后得到 C/A 码
ss_ca = g1.*g2;
ca = ss_ca;
ind1 = find(ca == -1);
ind2 = find(ca == 1);
ca(ind1) = ones(1,length(ind1));
ca(ind2) = zeros(1,length(ind2));
y = ca;

% 子函数 fGenerateNavigationData
function y = fGenerateNavigationData(x1,y1,z1);
% 将传进的参数转换为十六进制
x = x1;
y = y1;
z = z1;
str1 = num2hex(x);
str2 = num2hex(y);
str3 = num2hex(z);
Table1 = [0 0 0 0; 0 0 1 0; 0 0 1 1; 0 1 0 0; 0 1 0 1; 0 1 1 0; 0 1 1 1;
1 0 0 0; 1 0 0 1; 1 0 1 0; 1 0 1 1; 1 1 0 0; 1 1 0 1; 1 1 1 0; 1 1 1 1;];
TotalStr = [str1 str2 str3];
DataCode = zeros(1,1);
l = length(TotalStr);
% 将 TotalStr 中的数转换为 ASCII 码表中的数值
for i = 1:l
    temp = int8(TotalStr(i));
    if(temp>58)
        temp = temp-96+10;
    else
```



```
temp=temp-47;  
end  
  
DataCode = [DataCode Table1(temp,:)];  
end  
y = DataCode(2:length(DataCode));
```

8.3 本例小结

本例以 GPS 卫星为对象，基于 MATLAB 模拟其生成 C/A 码及导航电文的过程，通过该例读者可以了解和掌握基于 MATLAB 进行信号生成的方法及流程，对于 GPS 信号及导航电文特点及生成原理也将有所了解。

第三部分 电力电子仿真实例



引言——SimPowerSystems 简介

这一部分四个仿真实例对基于 MATLAB 进行电力电子对象建模与仿真相关内容进行介绍。MATLAB 提供了 SimPowerSystems 和 Simscape 这两个工具箱对电力电子对象进行设计与仿真，其中 SimPowerSystems 用于电力对象辅助设计仿真。而 Simscape 包含四个子工具箱 SimDriveline、SimElectronics、SimHydraulics 和 SimMechanics，分别用于传动系统、电子系统、液态/流体对象系统及机械系统设计与仿真。在引言中首先简单介绍基于 SimPowerSystems 设计电力系统的流程与特点。

一、SimPowerSystems 特点及分类

SimPowerSystems 提供了适合基本电气回路和具体电力系统的建模与仿真工具。这些工具可帮助对发电、输电和配电以及向机械能量转换的过程进行建模。SimPowerSystems 非常适合开发复杂的自给型电力系统，例如汽车、飞机、生产厂中的电力系统，以及公共电力设施应用。

SimPowerSystems 和 Simulink 一起为多域建模及控制器设计提供了一个高效环境。通过将仿真的电子部件连接到其他 Simulink 模块，可快速地绘制电路拓扑图，同时可分析回路与机械、热和控制系统的交互作用。

主要特点：

- (1) 允许采用标准符号对电子电路进行建模和仿真。
- (2) 提供全面的模块工具箱，以构建具体的电力系统模型。
- (3) 提供普通交流和直流电动驱动器的具体模型。
- (4) 利用 Simulink 求解器技术提供高精度的仿真。
- (5) 使用离散和相量仿真模式，加速模型执行并允许实时执行。
- (6) 提供分析方法以获得电路的状态空间表示，计算机器负荷流，并处理电流和电压。

SimPowerSystems 工具箱包含超过 130 个模块，这些模块分布在六个子工具箱中。可将该工具箱与 Simulink 配合，以创建由 SimPowerSystems 元件和控制回路组成的电路结构模块图。电路模块和 Simulink 模块之间的互连允许研究控制系统如何与电力系统



关联。

工具箱中还包含测量和控制电流与电压的模块。它们可作为电子信号与 Simulink 模块之间的链接。其他的测量模块可提供傅里叶分析和三相序列分析。可在模型中的选定位置使用测量模块，以便通过 Simulink 示波器观察电子信号，或将其转换成 Simulink 信号。

SimPowerSystems 工具箱中的子工具箱如图 C1 所示。

(1) Electrical Sources (电源) 包括交流和直流电压源、可控电压和可控电流。

(2) Electrical Circuit elements (电路元件) 包括电阻、电感电容分支和负载、线性和饱和变压器、避雷器和断路器以及输电线路模型。

(3) Electric Machinery (电动机械) 包括同步、永磁同步及直流机械模型、励磁系统、液压和汽轮机调速器系统模型。

(4) Power Electronics (电力电子) 包括二极管、简单和复杂可控硅、GTO、开关、MOSFET、IGBT 模型及通用电桥。

(5) Control and measurement (控制和测量) 包括电压、电流和阻抗测量、RMS 测量、有功功率和无功功率计算，以及计时器、万用表和傅里叶分析、HVDC 控制、总谐波失真和 abc-to-dq0 及 dq0-to-abc 变换。

(6) Three-phase components (三相元件) 包括 RLC 负载和分支、断路器和故障、 π 节线路、电压源、二极管、可控硅、变压器、发生器、分析器以及测量等。

(7) 专门应用工具箱，主要有：

- FACTS——柔性交流输电系统相量模型。
- Distributed Resources (分布式资源)——风力涡轮相量模型。
- Electric Drives(电气驱动器)——用于激活机械旋转动作的可编辑电气驱动器模型，其中包括每种驱动器中的马达、转换器和控制器的详细说明。Electric Drives 工具箱包括永磁式、同步和异步(感应)型。转换器和控制器可实现此类马达的最常见速度和转矩控制方式。

(8) Powergui 模块，基于 SimPowerSystems 建立的模型中要求有且仅有一个 Powergui 模块。

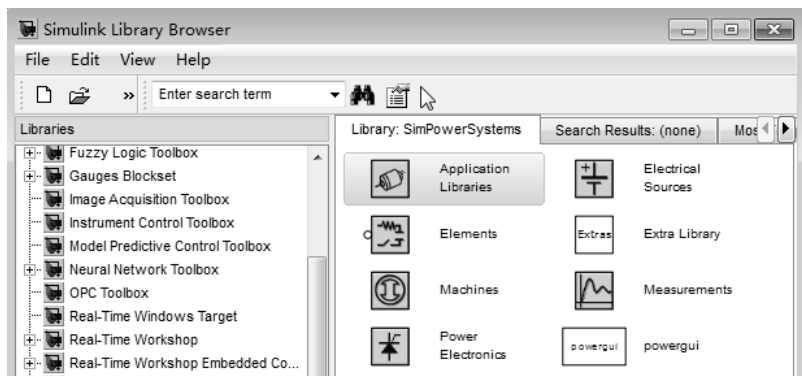
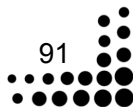


图 C1 SimPowerSystems 工具箱分类图





二、SimPowerSystems 求解器及仿真特点

1. 求解器特点

(1) Simulink 求解器非常适合任何规模的连续离散时间(模拟)、离散时间、混合以及混合信号仿真。它们可支持代数约束和状态事件,包括如工厂电力瞬间改变之类的间断情况。它们可提供快速、可靠和准确的仿真结果。

通过 SimPowerSystems,可在 Simulink 中使用变步长积分器来执行高度精确的电力系统模型仿真。其中一些积分器可处理在实际电力系统建模中常遇到的数值刚性系统。Simulink 提供的零点穿越检测功能,能以完全机器精度检测并求解不连续过程。

在 Simulink 中使用仿真加速器,可以提高仿真速度,最高能达到常规的仿真速度的 10 倍。

(2) SimPowerSystems 提供了两种可连续仿真电力系统的仿真模式:离散仿真和相量仿真。

离散仿真采用固定步长梯形积分法来仿真系统,特别适合带电力电子设备的电力系统模型。该模式还有利于模型的实时执行。

相量仿真则采用一组固定频率代数方程取代电力网络微分方程。相量仿真有利于进行多机器系统的暂态稳定性研究。

2. 仿真特点

(1) SimPowerSystems 提供了多种系统分析工具,其功能包括:

- 显示稳态电压和电流。
- 显示并修改初始状态值。
- 执行负荷流和机械初始化。
- 显示阻抗对比频率测量结果。
- 生成稳态计算报告。

(2) SimPowerSystems 图形用户界面可显示测量电流和电压及所有状态变量的稳态值,包括电感电流和电容电压。

负荷潮流计算引擎可计算同步和异步机器中的初始电流。该功能自动将结果初始电流写入到机器参数中。只需指定回路中要仿真的电压和功率值,然后单击按钮即可计算负荷潮流。

(3) SimPowerSystems 允许分析电路网络拓扑结构,并计算回路的等价状态空间模型,而不必运行仿真。可将状态空间模型链接到控制系统工具箱(单独提供)中的 LTI Viewer 接口,从而获得时域和频域响应。

三、SimPowerSystems 仿真示例

这里通过简单的例子来介绍基于 SimPowerSystems 建模过程。读者可按照以下步骤

逐步在自己的计算机中实现，以初步了解基于 SimPowerSystems 建模与仿真过程。

(1) 在 MATLAB 命令行中输入 “powerlib”，打开 SimPowerSystems (powerlib) 工具箱，如图 C2 所示。

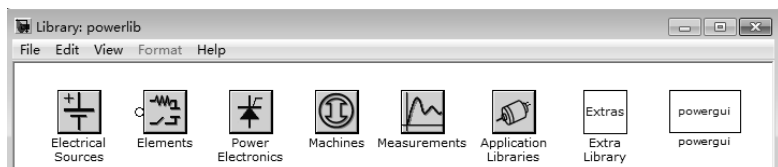


图 C2 powerlib工具箱分类图

(2) 在 powerlib 窗口中的 File 目录下，单击 “File|New|Model”，建立一个新的窗口来包含所要建立的电路，并将文件名存为 “circuit1”。

(3) 在 “Electrical Sources” 中选择 “AC Voltage Source” 模块，将其拖入 circuit1 中。

(4) 双击 “AC Voltage Source” 模块，设定参数为：424.4kV，0degrees，60Hz。

(5) 将 “AC Voltage Source” 模块名称修改为 Vs。

(6) 在 “Elements” 中选择 “Parallel RLC Branch” 模块，设置参数为：180.1Ω，26.525mH，117.84μF，并将名称修改为 “Z_eq”。

(7) 再复制一个 “Parallel RLC Branch” 模块，将其修改为电阻，参数为 2Ω。

注意：在 “Parallel RLC Branch” 模块参数设置窗口中，将 L 设为 inf，C 设为 0 并确定后，此时 “Parallel RLC Branch” 模块只剩下一个电阻。同样道理如果将 R 设为 0，则电阻部分将消失。

(8) 将电阻模型名称设为 “R_eq”。

(9) 在 “Elements” 选择 “Ground” 模块。此时用连线将部件连接起来，如图 C3 所示。

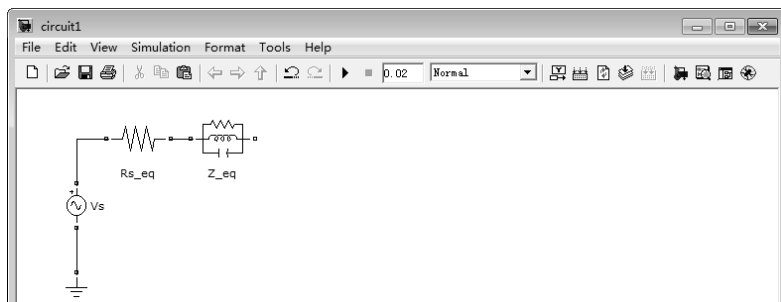


图 C3 SimPowerSystems 建模示例电路部分模块图

(10) 在 “Elements” 选择 “Pi Section Line” 模块，设置参数为：0.011Ω/km，0.8674mH，13.41nF。

(11) 在 “Elements” 选择 “Series RLC load” 模块，设置参数为：424.4e3，60Hz，110e6/300W，110e6vars，并将模块改名为 “110MVar”。

(12) 在 “Measurements” 中选择 “Voltage Measurement” 模块。

(13) 在 “Simulink” 中选择 “Gain” 模块，并设增益为 $K = \frac{1}{424.4 \times 10^3 \times \sqrt{2}}$ 。



(14) 在 powerlib 窗口根目录下选择 “powergui” 模块。

注意：“powergui” 模块对于基于 SimPowerSystems 工具箱仿真必需的，如果没有仿真，开始时会报错。

最终建立仿真模型如图 C4 所示，仿真结果如图 C5 所示。

注意：由图 C4 可看出，Simulink 模块和 SimPowerSystems 模块之间连接需要通过敏感器之类的模块来实现信号转换。

前面建立的模型定义为模式一，下面对模式一进行适当的修改建立模式二并进行仿真。

(1) 在模式一中加入开关 “Breaker”，开关模型可在 “Elements” 中找到，双击打开对话框，设置参数为： 0.001Ω ，0，inf，0， $[(1/60)4]$ 。

(2) 将模型名称改为 “circuit2”，如图 C6 所示。

(3) 双击打开 “Pi Section Line”，将 “number of sections” 设置为 1，仿真并保存结果至工作空间。

(4) 将 “number of sections” 设置为 10，仿真并保存结果至工作空间。

(5) 在工作空间保存的两组结果放在一个图中，如图 C7 所示。

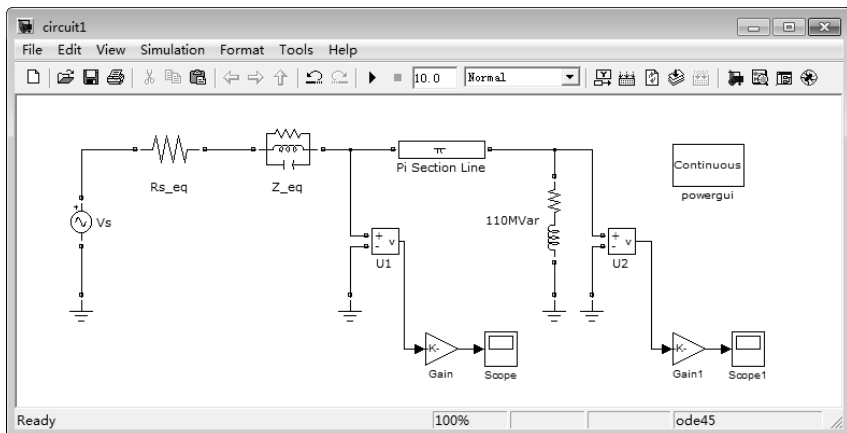
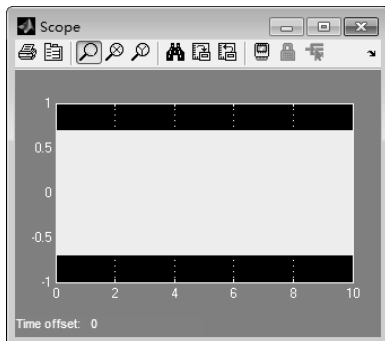
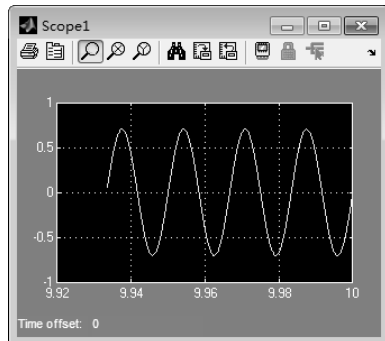


图 C4 SimPowerSystems 示例电路模式一



(a)



(b)

图 C5 SimPowerSystems 示例电路模式一仿真结果

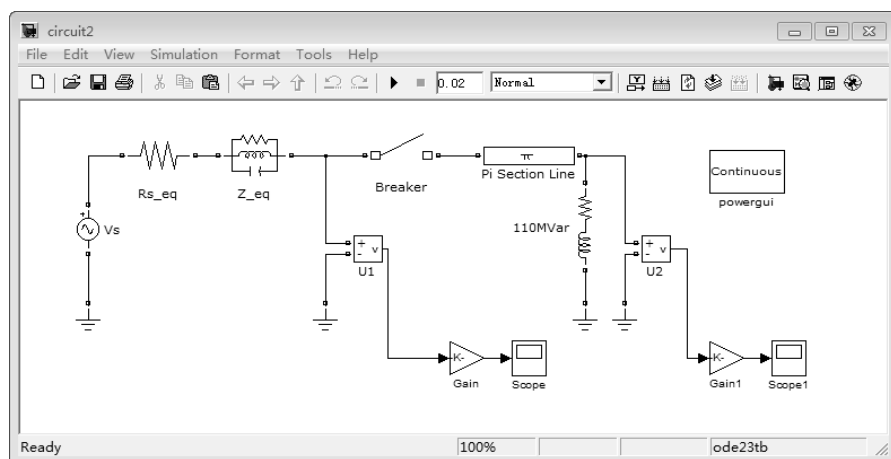


图 C6 SimPowerSystems 示例电路模式二

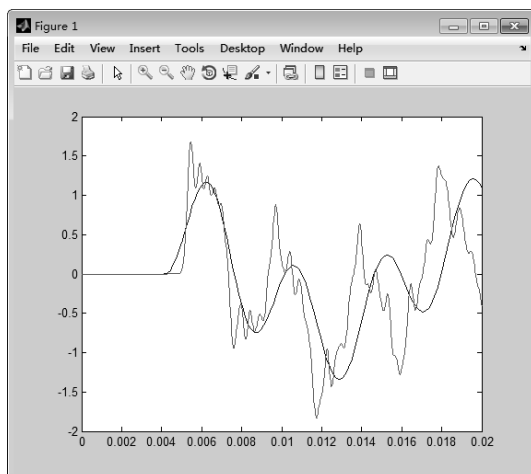


图 C7 SimPowerSystems 示例电路模式二仿真结果



第 9 例 燃料电池汽车动力系统仿真

石油属于非可再生能源，随着其消耗日益增多，世界各国纷纷展开其他替代能源及相关技术的研究，燃料电池汽车技术研究是其中一个重要的项目，引起众多研究机构的关注。燃料电池汽车是由燃料电池及蓄电池提供能量的电机驱动，其系统属于电力系统和传动系统的混合对象。其中电机驱动及汽车惯性属于传动系统部分，而燃料电池及蓄电池属于电力系统部分。

MATLAB 为研究燃料电池汽车相关技术研究准备了良好的研究条件，其中 SimDriveline 工具箱能够很好地支持传动系统设计与仿真，而 SimPowerSystems 工具箱则能很好地支持电力系统方面的研究，更为关键的是 SimPowerSystems 工具箱和 SimDriveline 工具箱基于统一的 Simulink 底层框架，因此两者可实现无缝连接，从而实现多领域仿真。这就使得 MATLAB 为燃料电池汽车研究提供了一个相对于其他软件或工具更为科学、准确和全面的研究平台。

由于之前已经介绍过 SimPowerSystems 工具箱相关知识，因此本例中首先介绍 SimDriveline 工具箱相关知识，其次通过一个燃料电池汽车能量分配方案展示基于 SimPowerSystems 和 SimDriveline 联合设计多领域仿真的流程及特点。

通过本例学习，需要掌握以下几个方面：

- 了解 SimDriveline 工具箱的特点及组成。
- 掌握基于 SimDriveline 设计传动系统的方法和流程。
- 掌握基于 SimPowerSystems 设计电力系统的方法和流程。
- 掌握基于 SimPowerSystems 和 SimDriveline 混合设计多领域系统方法和流程。

9.1 SimDriveline 简介

9.1.1 SimDriveline 功能概述

SimDriveline 为传动系统（驱动系统）的力学建模与仿真提供有力的工具。这些工具包括像齿轮、转动轴和离合器等部件，标准的变速器模板，发动机和轮胎模型。SimDriveline 专门为传动系的力学分析进行了易用性和计算速度方面的优化。它实现了与 MathWorks 控制系统设计和代码生成产品的集成，这样不仅可以进行控制器设计，而且还能够把机械系统模型生成实时代码，在实时环境中对控制器进行测试。



SimDriveline 可以广泛用于汽车、航空、国防和工业领域。它尤其适合于汽车和航空传动系统的控制器开发。

它的特点有以下几点。

(1) 在 Simulink 下对传动系力学进行定义的建模环境。

SimDriveline 为在 Simulink 环境中建立传动系模型提供了有效的途径。用户可以使用模块图网络描述来表示一个系统。不同的模块代表不同的部件,如齿轮、离合器和液力变矩器。连接不同模块之间的线代表旋转部件,如驱动轴。在 SimDriveline 中,用户可以拥有 Simulink 的所有功能。使用传感器模块,用户可以测量速度、加速度和转矩,并且把这些测量信号值传给标准的 Simulink 模块。Simulink 信号能够通过执行器模块对驱动转矩进行定义,或者预先设定传动轴的动力学参数。SimDriveline 为实现完全的机械系统 3-D 仿真器提供了另外一条有效的途径,它完全专注于旋转机械的力学仿真。每一根杆件的运动被限制于绕某个轴的转动,用户可以通过一个简单的惯性质量部件为每根杆件进行质量参数赋值。只对每根杆件的旋转速度进行记录的结果就是加快仿真执行的速度。

(2) 通用的齿轮结构工具箱。

SimDriveline 包括了很多部件的模块工具箱,这些模块定义了连接轴之间的部件的运动和转矩关系。

SimDriveline 能够让用户开发简单和复杂的齿轮系统。预先定义的模块提供了基本的齿轮结构。用户可以通过对齿轮传动比进行合适的赋值,从而实现对这些模块的参数化设置。

用户可以把这些简单的结构进行组合,从而生成更复杂的齿轮系统,例如在很多变速器中看到的那样。SimDriveline 也提供了通用的复杂齿轮系统的模型,例如 Ravigneaux 齿轮组和差速器。

(3) 动态元件工具箱,包括离合器和转动限位器 (Rotational stops)、液力变矩器和扭转的弹簧-减震器。

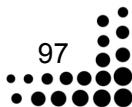
SimDriveline 工具箱中包括了很多动力学元件和齿轮组,从而提供了一套完整的传动轴连接件模型。例如,可控摩擦离合器模型 (Controllable Friction Clutch) 实现了从离合器接合到具有不同转速的轴锁止的过程的建模。可控摩擦离合器模型 (Controllable Friction Clutch) 针对过渡过程采用了一个动态摩擦模型,在锁止条件下采用了一个静态摩擦模型。

SimDriveline 也包含了以下动态元件:

- 液力变矩器——让用户对流体轴承连接进行建模。
- 硬限位器 (Hard Stop) ——让用户对传动系中的间隙 (backlash) 进行建模,并且考虑其造成的功率损失。
- 扭转弹簧-减震器——让用户对具有间隙 (backlash) 特性的扭转减震弹簧进行建模。

(4) 车辆部件的基本模型,包括发动机、纵向车辆动力学和轮胎。

SimDriveline 提供了 Lepelletier 6 挡和 7 挡变速器, Simpson 的 4 挡和 Ravigneaux 的





4 挡变速器。这些子系统可以作为模板使用，为用户建立变速器系统模型提供指导。用户可以通过模型中的部件模块来修改模型的拓扑结构或者传动比、轴的惯性质量和离合器特性等参数。

SimDriveline 包括简单的车辆部件模型，让用户对一个完整的动力系统及其对车辆运动性能的影响进行建模。这些模型能够对整个系统的性能进行早期评价。对于变速器模板而言，用户可以添加动力系统和车辆动力学模型所需要的细节内容。

SimDriveline 中包括的简单车辆部件模型工具箱包括以下部分：

- 汽油机和柴油机模块——对转矩、发动机转速和油门开度之间的关系进行建模，其油门开度值是通过 Simulink 输入信号来赋值的。
- 轮胎模型——对轮胎-道路之间相互作用产生的各种力和力矩进行求解，并且计算轮胎和车轮的旋转速度。
- 车辆纵向动力学——对传动系的转矩和加速度引起的车辆响应进行评价。

9.1.2 SimDriveline 工具箱分类

有两种方式打开 SimDriveline 工具箱。

(1) 首先打开 Simulink，然后在左侧的列表中找到 Simscape 并单击，展开后可看到 SimDriveline，再次单击，此时在右侧会显示 SimDriveline 工具箱中的模块分类。

(2) 在 MATLAB 命令行中输入 drivelib，会直接弹出 SimDriveline 工具箱。

以第一种方式打开 SimDriveline 工具箱，如图 9-1 所示。

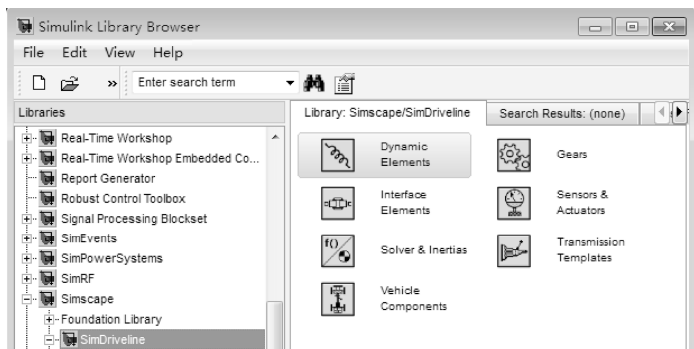


图 9-1 SimDriveline 工具箱分类图

由图可看出，SimDriveline 工具箱中模块分为七大类。

(1) 动力学模块工具箱。用于模拟如离合器、扭矩转换器、阻尼弹簧等传动部件，这些部件产生驱动力矩。

(2) 齿轮模块工具箱。用于模拟简单和复杂等多种齿轮，能够满足多种不同传动系统约束。

(3) 接口工具箱。用于连接 Simscape mechanical 旋转模块。

(4) 敏感器和执行机构工具箱。用于模拟观测传动力矩位置、力矩敏感器以及驱动



传动机构运动的执行机构。

(5) 求解与惯性模块工具箱。用于模拟旋转物体的转动惯量。此外，该工具箱中还包含传动仿真所需的环境模块。

(6) 传动模板工具箱。该工具箱中已集成了一些动力学、齿轮、敏感器和执行机构等模块，用户可在这些模板的基础上定制自己所需的特定传动系统。

(7) 汽车组件工具箱。包含整车动力系统部件，包括发动机模型、轮式车辆等。

9.1.3 基于 SimDriveline 建模特点

基于 SimDriveline 建立的传动系统需要注意以下几点。

(1) SimDriveline 工具箱模块之间连接的接口有两种类型，一种与 Simulink 模块端口一样，另一种则不同，这两者模块接口不能连接。

(2) Simulink 中的连接线没有任何物理信息和数学运算，仅仅表示两者之间的连接关系。而 SimDriveline 中的连接线传递了轴及力等物理信息。

(3) SimDriveline 模块传递时遵循无能量损耗原则，因此模块间的力矩与能量传递是相等的。

(4) SimDriveline 模块中所有端口都必须连接到另一个端口。

(5) SimDriveline 模块间连接时需要遵守角速度相同原则，因此两个连接模块间初始角速度必须相同。

9.2 燃料电池汽车仿真

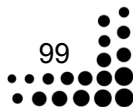
9.2.1 燃料电池汽车简介

燃料电池汽车的工作原理是，使作为燃料的氢在汽车搭载的燃料电池中，与大气中的氧发生化学反应，从而产生出电能启动电动机，进而驱动汽车。甲醇、天然气和汽油也可以替代氢（从这些物质里间接地提取氢），不过将会产生极度少的二氧化碳和氮氧化物。但总的来说，这类化学反应除了电能就只产生水。因此燃料电池车被称为“地道的环保车”。

9.2.2 燃料电池汽车仿真电路设计

本例是一个燃料电池汽车动力系统仿真例子，基于 MATLAB 中电力工具箱 (SimPowerSystems) 和传动工具箱 (SimDriveline) 完成，属于多领域仿真。建立的仿真模型如图 9-2 所示。

燃料电池汽车是由燃料电池及蓄电池提供能量的电机驱动的，由四部分组成：电动机、电池、燃料电池和 DC/DC 转换器。



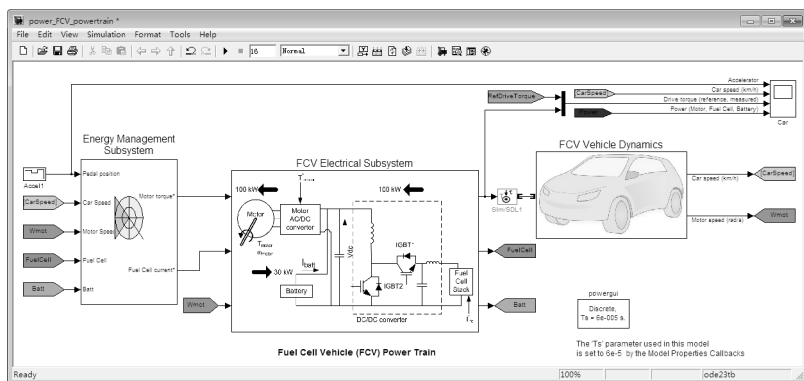


图 9-2 燃料电池汽车仿真整体框图

电动机、蓄电池及燃料电池如图 9-3 所示。电动机由一个 288V、100kW 的永磁同步电机及驱动电路组成。电动机为 8 极磁铁内凹型，通过弱磁矢量控制可使得电机最高转速达到 12500 转每分钟。蓄电池为 13.9Ah、288V、25kW 的锂离子电池。燃料电池为一个具有 400 个燃料元胞、288V、100kW 的质子交换膜燃料电池栈。DC/DC 转换器由电流整形。其中电机驱动如图 9-4 所示。

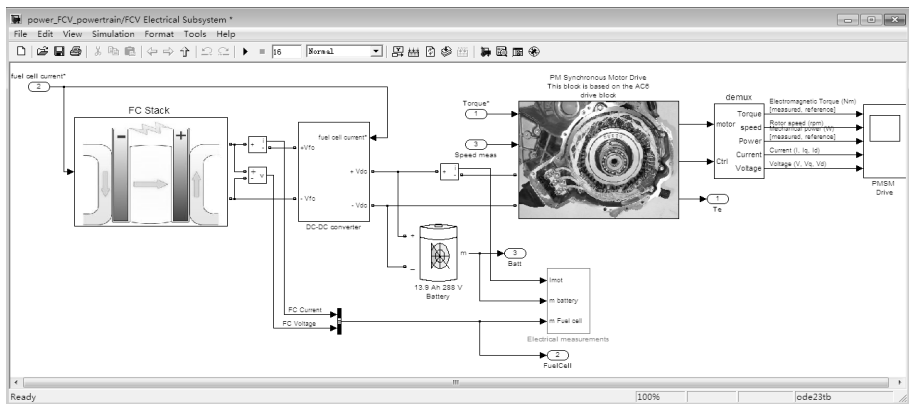


图 9-3 发电系统

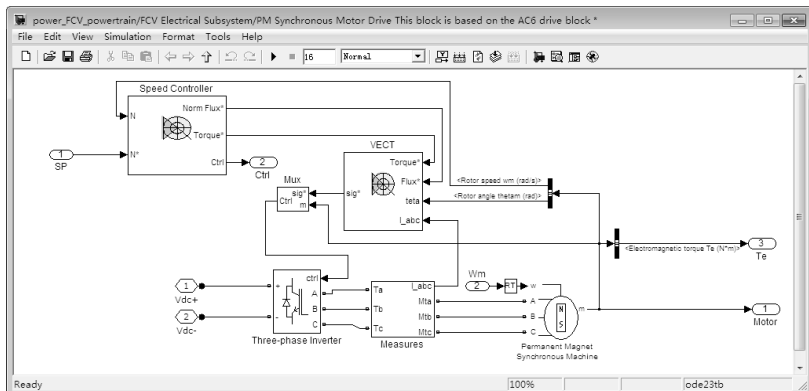


图 9-4 同步电机驱动

燃料电池汽车动力学系统如图 9-5 所示，包含以下几个部分。

- (1) 单齿轮减速装置，用于降低电动机传递来的转速以便增加力矩。
- (2) 差分器，用于将输入力矩分成两个相等的力矩。
- (3) 轮胎模型，用于模拟汽车与地面作用动力学。
- (4) 汽车动力学，用于模拟汽车的运动。
- (5) 黏性摩擦模型，用于模拟机械系统的损耗。

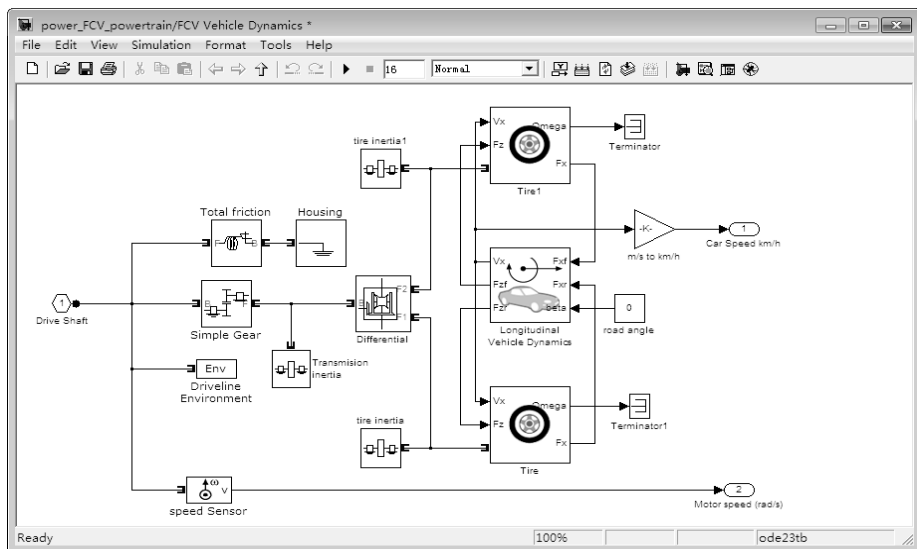


图 9-5 汽车动力学及传动框图

能量管理子系统 (EMS) 如图 9-6 所示。设定了电机驱动、燃料电池系统及 DC/DC 转换器等参考信号，以准确地分配由两个电源提供的电能。这些参考信号主要由加速器位置及测量得到的燃料电池汽车速度计算得到，介于-100%和 100%之间。需要注意的是负的加速器位置代表了正的刹车位置。

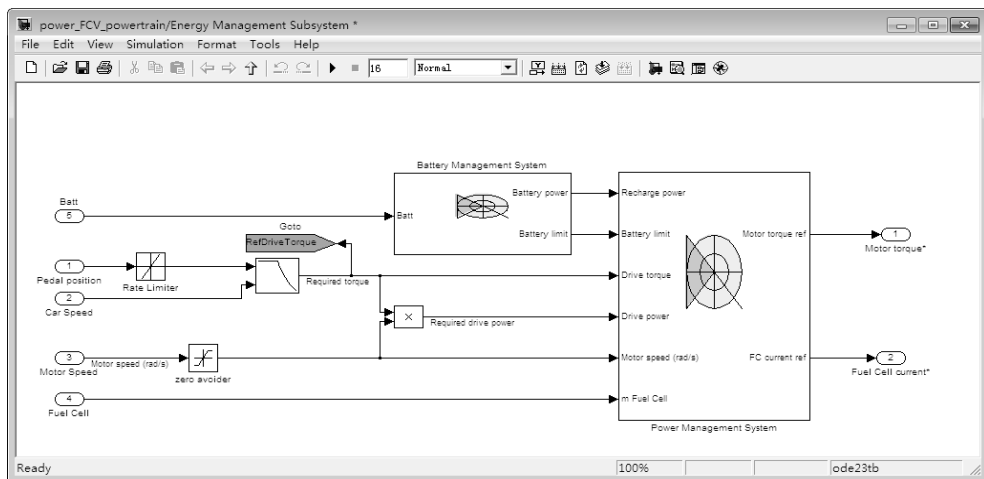


图 9-6 能量管理子系统



蓄电池管理系统保持充电率在 40%至 80%之间。同时它通过控制用电负载保证蓄电池在供电时,母线电压保持稳定。能量管理系统首先将汽车能量需求建立为燃料电池和蓄电池现有电量的函数,然后根据这个函数设定电机的参考能量。能量通过 DC/DC 转换器电流控制。

仿真结果主要是观测四部分变量。

(1) 第一个是加速器位置,汽车速度,驱动力矩和能量的变化。

(2) 第二个是燃料电池汽车电力子系统相关变量,可观测电机电力矩,电机转速,机械能量,定子电流和定子电压等。

(3) 第三个是燃料电池及蓄电池相关变量,可观测燃料电池的电压和电流,蓄电池荷电状态,DC/DC 转换器状态。

(4) 第四个是电力参考能量信号。

9.2.3 仿真结果及分析

该仿真模拟了燃料电池汽车一个完整的周期内的变化:加速、巡航,以及在加速和刹车时对蓄电池进行充电。仿真开始后一分钟,汽车开始加速。汽车速度在 12 秒内由 0 加速到 90 千米每小时,最后在 16 秒之内降到 80 千米每小时。最后的 16 秒之内,前 4 秒内将加速踏板设为常量的 70%,在下 4 秒内将加速踏板设为常量的 25%,然后在接下来的 4 秒内将加速踏板设为常量的 85%,最后的 4 秒内将加速踏板设为常量的-70% (刹车)。

仿真结果如图 9-7 所示,由图可见整个仿真过程如下。

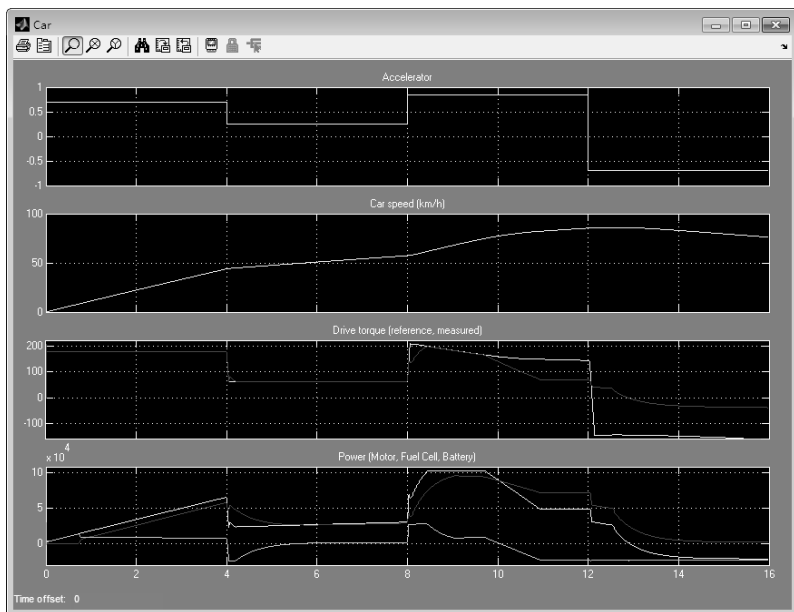


图 9-7 仿真结果



(1) 在第 0 秒, 燃料电池汽车停止, 加速踏板设为常量的 70%, 蓄电池提供能量直到燃料电池开始工作。

(2) 在第 0.7 秒, 燃料电池开始提供能量, 不过由于燃料电池较大的时间特性, 刚开始还未能达到所需的参考能量, 所以此时仍然由蓄电池供电。

(3) 在第 4 秒, 加速踏板降为常量的 25%。燃料电池供能不能马上降下来, 此时蓄电池开始充电以保持所需的力矩。

(4) 在第 6 秒, 燃料电池达到参考能量设定值, 蓄电池不再工作。

(5) 在第 8 秒, 加速踏板设定为常量的 85%, 蓄电池与燃料电池一起工作, 承担 25kW 的功率消耗。

(6) 在第 8.05 秒, 蓄电池与燃料电池提供的总能量未能达到要求的数值, 因此此时测量的力矩未能达到参考值。

(7) 在第 8.45 秒, 测量的力矩值达到参考值。燃料电池能够提供的能量持续上升, 因此蓄电池提供的能量逐渐降至 6kW。

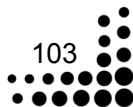
(8) 在第 10.9 秒, 蓄电池电量小于 40% (仿真初始时为 40.32%), 因此蓄电池需要充电, 燃料电池为电机及蓄电池提供能量。此时要求的力矩无法达到。

(9) 在第 12 秒, 加速踏板设为 -70%, 此时电机可看作是被汽车轮胎驱动, 机械能转换为蓄电池化学能。此时要求的 $-140\text{N} \cdot \text{m}$ 力矩也未能达到, 因为蓄电池只能吸收 25kW 的能量。

注意: ①系统已离散化, 时间常量为 $60\mu\text{s}$ 。②PMSM 驱动模块及 DC/DC 转换器模块采用平均值, 这样可以采用较大的仿真步长。

9.3 本例小结

本例以燃料电池汽车为对象, 基于 SimPowerSystems 和 SimDriveline 设计燃料电池的仿真系统, 通过该例读者可以了解和掌握基于 SimPowerSystems 和 SimDriveline 建模、仿真的特点及流程, 对于燃料电池汽车结构和系统组成也将有所了解。





第 10 例 船舶雷达系统 射频前端电路分析

本例是针对第 6 例中船舶雷达系统中射频前端电路的建模与分析。现在电子系统中除了射频前端电路，大部分都能用数字电路代替模拟电路，可见其特殊性。MATLAB 提供了射频（RF）工具箱来辅助射频系统的设计与分析。本例首先介绍 RF 工具箱的特点与建模过程，其次对基于 RF 工具箱设计船舶雷达系统中射频前端电路进行分析。

通过本例学习需要了解和掌握以下几个方面：

- 了解和掌握基于 M 语言的 RF 工具箱数据类型及相应的函数。
- 掌握基于 Simulink 工具箱分类及特点。
- 掌握基于 RF 工具箱设计与仿真射频电路的方法与过程。

10.1 RF 工具箱简介

10.1.1 基于 M 语言的 RF 工具箱特点及仿真过程

射频工具箱能够对射频电路中的滤波、传输、放大和混频等电路及传输过程进行建模与仿真，能够建立射频部件之间的互连网络，并进行设计、分析及观测射频部件及网络。这里介绍射频工具箱中两个重要的概念：S 参数和射频对象数据格式。

1. S 参数

微波系统主要研究信号和能量两大问题：信号问题主要是研究幅频和相频特性；能量问题主要是研究能量如何有效地传输。微波系统是分布参数电路，必须采用场分析法，但场分析法过于复杂，因此需要一种简化的分析方法。微波网络法被广泛运用于微波系统的分析，是一种等效电路法，在分析场分布的基础上，用路的方法将微波元件等效为电抗或电阻器件，将实际的导波传输系统等效为传输线，从而将实际的微波系统简化为微波网络，把场的问题转化为路的问题来解决。微波网络理论是在低频网络理论的基础上发展起来的，低频电路分析是微波电路分析的一个特殊情况。一般地，对于一个网络有 Y、Z 和 S 参数可用来测量和分析，Y 称为导纳参数，Z 称为阻抗参数，S 称为散射参数；前两个参数主要用于集总电路，Z 和 Y 参数对于集总参数电路分析非常有效，各参数可以很方便地测试；但是在微波系统中，由于确定非 TEM 波电压、电流非常困难，而



且在微波频率测量电压和电流也存在实际困难。因此，在处理高频网络时，等效电压和电流以及有关的阻抗和导纳参数变得较抽象。与直接测量入射、反射及传输波概念更加一致的表示是散射参数，即 S 参数矩阵，它更适用于分布参数电路。S 参数就是建立在入射波、反射波关系基础上的网络参数，适于微波电路分析，以器件端口的反射信号以及从该端口传向另一端口的信号来描述电路网络。同 N 端口网络的阻抗和导纳矩阵那样，用散射矩阵也能对 N 端口网络进行完善的描述。阻抗和导纳矩阵反映了端口的总电压和电流的关系，而散射矩阵是反映端口的入射电压波和反射电压波的关系。散射参量可以直接用网络分析仪测量得到，可以用网络分析技术来计算。只要知道网络的散射参量，就可以将它变换成其他矩阵参量。

一个典型的二端口网络如图 10-1 所示。下面以二端口网络为例说明各个 S 参数的含义，如上图所示。二端口网络有 4 个 S 参数， S_{ij} 代表的意思是能量从 j 口注入，在 i 口测得的能量，如 S_{11} 定义为从 Port1 口反射的能量与输入能量比值的平方根，也经常被简化为等效反射电压和等效入射电压的比值，各参数的物理含义和特殊网络的特性如下：

S_{11} ：端口 2 匹配时，端口 1 的反射系数。

S_{22} ：端口 1 匹配时，端口 2 的反射系数。

S_{12} ：端口 1 匹配时，端口 2 到端口 1 的反向传输系数。

S_{21} ：端口 2 匹配时，端口 1 到端口 2 的正向传输系数。

对于互易网络，有： $S_{12}=S_{21}$ 。

对于对称网络，有： $S_{11}=S_{22}$ 。

对于无耗网络，有： $(S_{11})^2 + (S_{21})^2 = 1$ 。

经常用到的单根传输线，或一个过孔，就可以等效成一个二端口网络，一端接输入信号，另一端接输出信号，如果以 Port1 作为信号的输入端口，Port2 作为信号的输出端口，那么 S_{11} 表示的就是回波损耗，即有多少能量被反射回源端 (Port1)，这个值越小越好，一般建议 $S_{11} < 0.1$ ，即 -20dB； S_{21} 表示插入损耗，也就是有多少能量被传输到目的端 (Port2) 了，这个值越大越好，理想值是 1，即 0dB， S_{21} 越大传输的效率越高，一般建议 $S_{21} > 0.7$ ，即 -3dB。

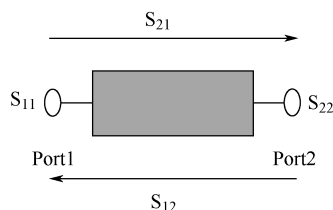
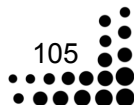


图 10-1 二端口网络的 S 参数

在 MATLAB 中通过定义矩阵符号来设置 S 参数值。以定义一个 50Ω 两端口 S 参数为例，介绍在 MATLAB 中如何定义 S 参数：

```
s11 = 0.61*exp(j*165/180*pi);
s21 = 3.72*exp(j*59/180*pi);
s12 = 0.05*exp(j*42/180*pi);
```





```
s22 = 0.45*exp(j*(-48/180)*pi);  
s_params = [s11 s12; s21 s22];
```

s_params 为建立的二端口 s 参数，其中 s11 为端口 1 的反射系数，s21 为端口 1 至端口 2 的传输系数，s12 为端口 2 至端口 1 的传输系数，s22 为端口 2 的反射系数。

射频工具箱同时支持三维 s 参数，其中第三维描述对应不同频率的 s 参数值。

2. 射频对象

射频工具箱用对象这一数据类型来表示射频部件及网络。通过对象构造器来建立对象。每一个对象都有预先定义的对象属性。属性定义了对应的特征。每一个属性都被赋予特定的值。每一个对象同时有一系列函数，这些函数可作用于对象上面。

表 10-1 列出了射频工具箱中三类常用的对象：射频数据对象、射频电路对象和射频模型对象。

10-1 射频工具箱三类常用对象

对象类型	名称	描述
射频数据对象	rfdata	存储射频数据，用于作图及网络数据转换
射频电路对象	rfckt	模拟射频对象，并通过参数设置和物理属性进行频域仿真
射频模型对象	rfmodel	模拟射频对象，并进行时域仿真和输出模型

下面对这三类对象具体的子对象进行介绍。

(1) rfdata 具体的子对象有以下几种。

- rfdata.data 包含网络参数数据的对象。
- rfdata.ip3 包含 IP3 信息的对象。
- rfdata.mixerspurs 包含混频信息的对象。
- rfdata.network 包含网络参数信息的对象。
- rfdata.nf 包含噪声图信息的对象。
- rfdata.noise 包含噪声信息的对象。
- rfdata.power 包含能量和相位信息的对象。

能作用于上述部分对象的函数有：

- extract 能够作用于 rfdata.data 和 rfdata.network 这两种对象上面，函数的作用为从特定对象上提取网络参数并以数组的形式返回提取结果。
- read 读取 rfdata.data 该对象的参数。
- write 将数据写入 rfdata.data 对象。

(2) rfckt 对象部分子对象有以下几种。

- rfckt.amplifier 运算放大器对象。
- rfckt.cascade 级联网络对象。
- rfckt.coaxial 同轴传输信道对象。
- rfckt.datafile 通用电路对象。
- rfckt.delay 延迟信道。



- rfckt.lcbandpasspi LC 带通 π 型网络。
- rfckt.lcbandpasstee LC 带通 T 型网络。
- rfckt.lcbandstoppi LC 带阻 π 型网络。
- rfckt.lcbandstoptee LC 带阻 T 型网络。
- rfckt.microstrip 微波信道。
- rfckt.mixer 混频器。
- rfckt.txline 通用信道模型。

同样也有一些相应的处理函数：

- analyze 在频域分析电路，适用全部 rfckt 子对象。
- copy 复制电路或数据对象，适用全部 rfckt 子对象。
- extract 提取参数，适用全部 rfckt 子对象。
- plot 在 X-Y 平面内画出指定对象的参数。
- write 在一个对象中写入数据。

(3) rfmodel 的子对象有以下几种。

- rfmodel.rational 有理函数模型。

处理的函数有：

- freqresp 计算模型频域响应。
- timeresp 计算模型时域响应。
- writeva 将模型数据写入文件。

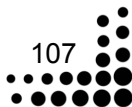
基于上述对象分类，生成具体对象的语句格式是：

`h = objecttype.objectname`

其中 h 为新对象的句柄，objecttype 为对象类型 (rfdata、rfckt 和 rfmodel)，objectname 为对象名称。

这里举个简单的例子说明，仿真结果如图 10-2 所示。

```
% 首先建立两个信道和一个放大器，并采用默认值
FirstCkt = rfckt.txline;
SecondCkt = rfckt.amplifier;
ThirdCkt = rfckt.txline;
% 设置两个信道的值
set(FirstCkt, 'LineLength', 0.001)
set(ThirdCkt, 'LineLength', 0.025, 'PV', 2.0e8)
% 设置放大器的值
read(SecondCkt, 'default.amp');
set(SecondCkt, 'IntpType', 'cubic')
% 通过作图验证射频性能
figure
lineseries1 = smith(SecondCkt, 'S11', 'S22');
set(lineseries1(1), 'LineStyle', '-', 'LineWidth', 1);
set(lineseries1(2), 'LineStyle', ':', 'LineWidth', 1);
legend show
```



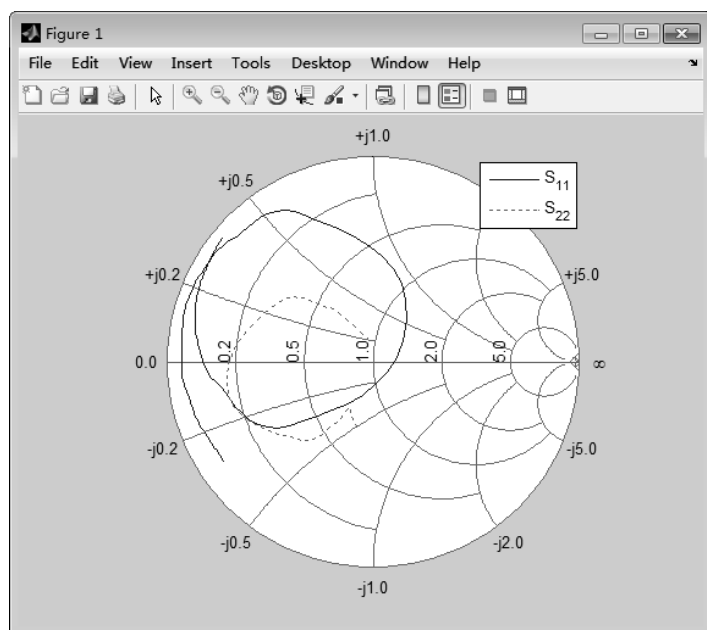


图 10-2 放大器性能

10.1.2 基于 Simulink 的 RF 工具箱分类

基于 Simulink 的 RF 工具箱能够实现与基于 M 语言的 RF 工具箱同样的功能。下面对基于 Simulink 的 RF 工具箱模块分类予以介绍。

图 10-3 所示为基于 Simulink 的 RF 工具箱模块分类。从图中可看出，主要有三类：元素子工具箱(Elements)、信号源子工具箱(Sources)和常用电路网络子工具箱(Utilities)。

图 10-4 所示为元素子工具箱内部图，该子工具箱内部为一些基本的电路单元，读者可基于这些基本的电路单元搭建射频电路，不过效率较低，一般采用常用电路网络子工具箱。

图 10-5 所示为信号源子工具箱，其作用是为信号传播系统信号源及传递过程附加各种噪声。

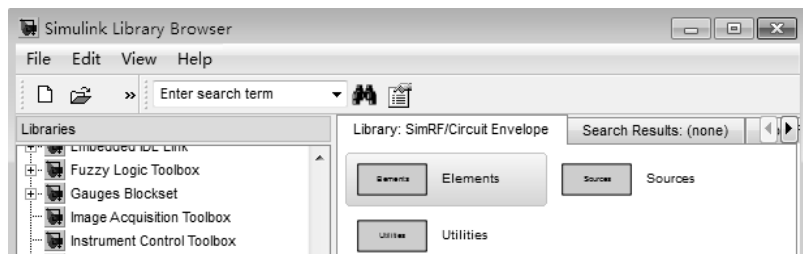


图 10-3 基于 Simulink 的 RF 工具箱模块分类

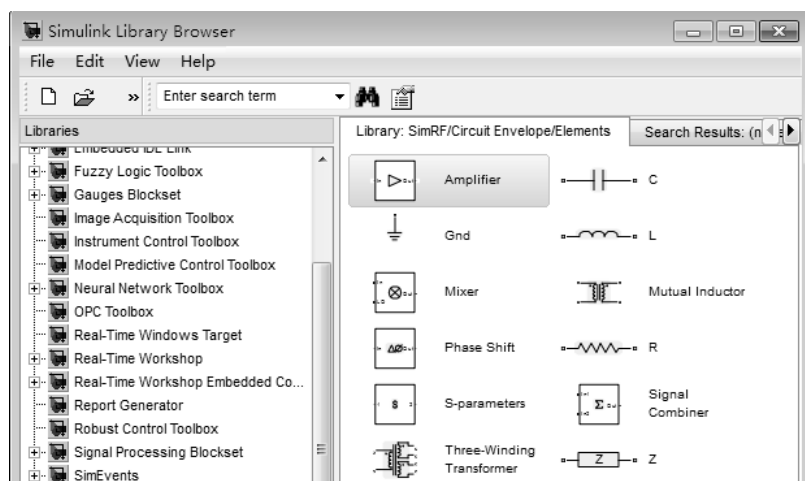


图 10-4 元素子工具箱

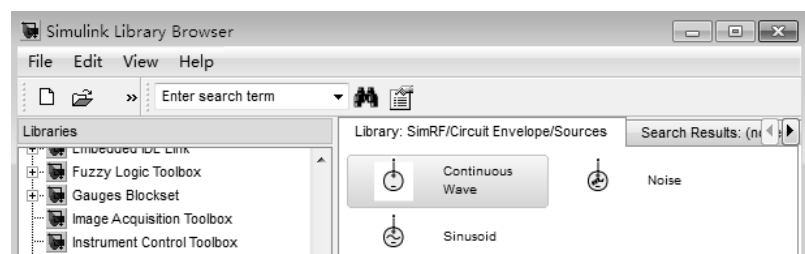


图 10-5 信号源子工具箱

图 10-6 所示为常用电路网络子工具箱内部图。从图中可看出该子工具箱还有一些更小的分类，这些类别中主要有以下七类。

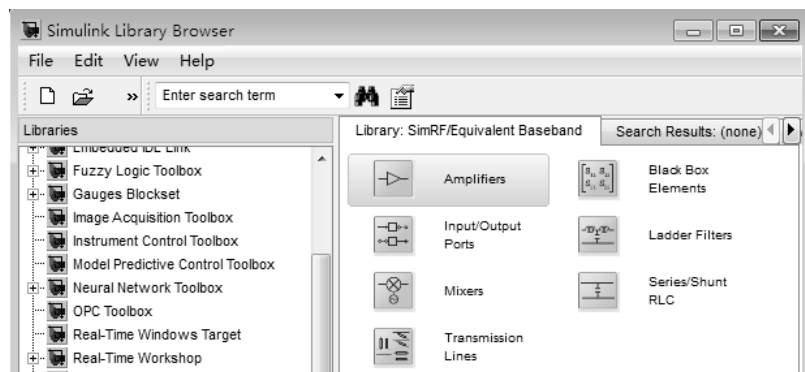


图 10-6 常用电路网络子工具箱

- (1) 放大器，如图 10-7 所示。
- (2) 黑盒网络电路，如图 10-8 所示。
- (3) 输入/输出端口，如图 10-9 所示。
- (4) 滤波器，如图 10-10 所示。



- (5) 混频器，如图 10-11 所示。
- (6) 并联/串联 RLC 电路。
- (7) 信道。

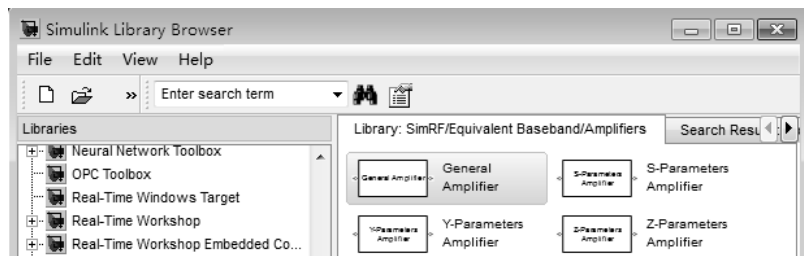


图 10-7 放大器

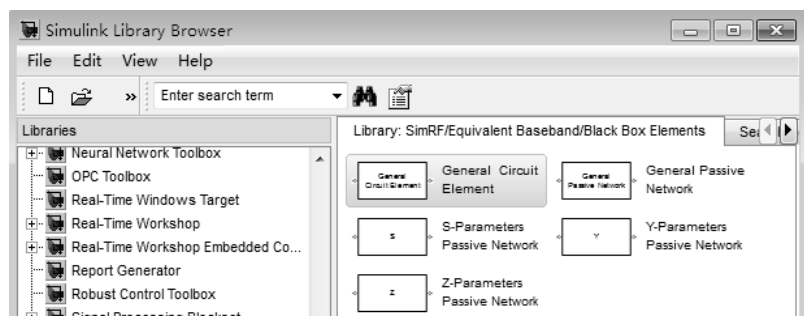


图 10-8 黑盒网络电路

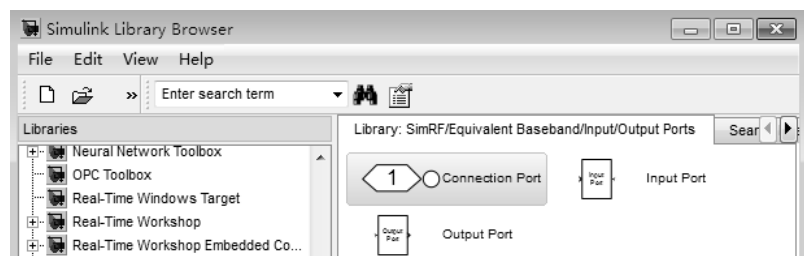


图 10-9 输入/输出端口

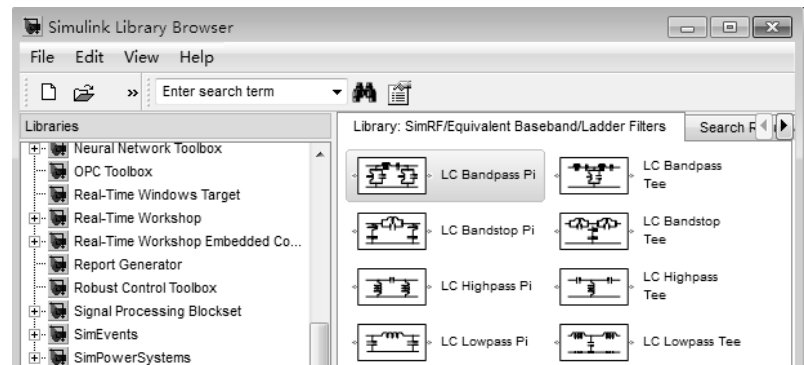


图 10-10 滤波器

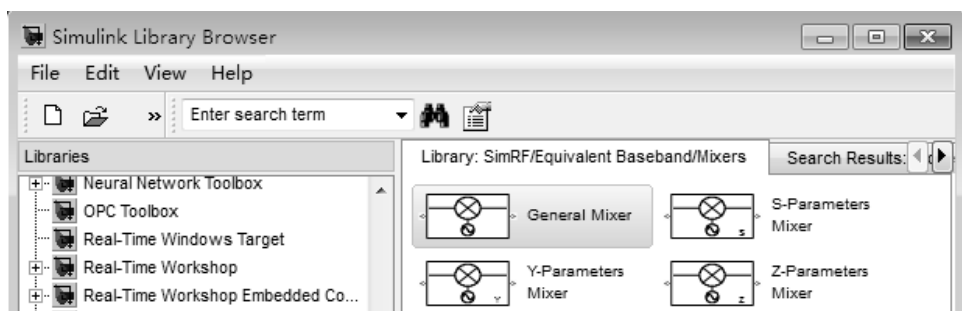


图 10-11 混频器

10.2 雷达射频前端电路设计与仿真

该例是第 6 例中关于电路的部分，以雷达射频前端电路为例介绍基于 SimRF 进行建模仿真过程。如图 10-12 所示为发射端电路，有混频器、带通滤波器、放大器，发射端电路出去后为天线模型。其建立过程如下所示。

- (1) 输入端口“Input Port”可从图 10-9 中得到。
- (2) 混频器“S-Parameters Mixer”可从图 10-11 中得到。
- (3) 滤波器“LC Bandpass Pi”可从图 10-10 中得到。
- (4) 输出端口“Output Port”可从图 10-9 中得到。

如图 10-13 所示为接收端电路，有一个两端口网络电路、带通滤波器、解频器和放大器，接收端电路之后是信号处理部分。其建模可参见发射段端口建立过程。

仿真结果如图 10-14 所示，由图可见，接收端输入增益为-146.2，而接收端输出增益为-111.2，说明接收端电路使得信号增益为 30dB。

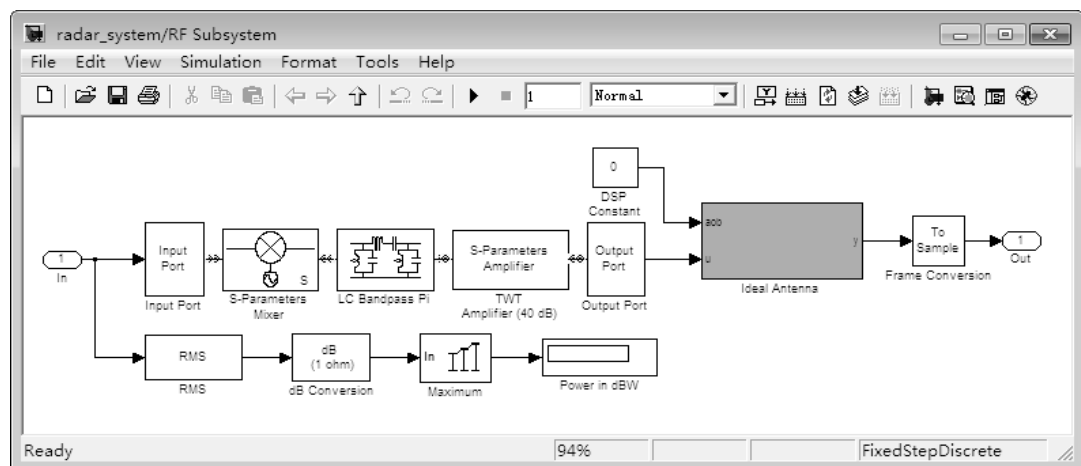


图 10-12 雷达射频发射端电路

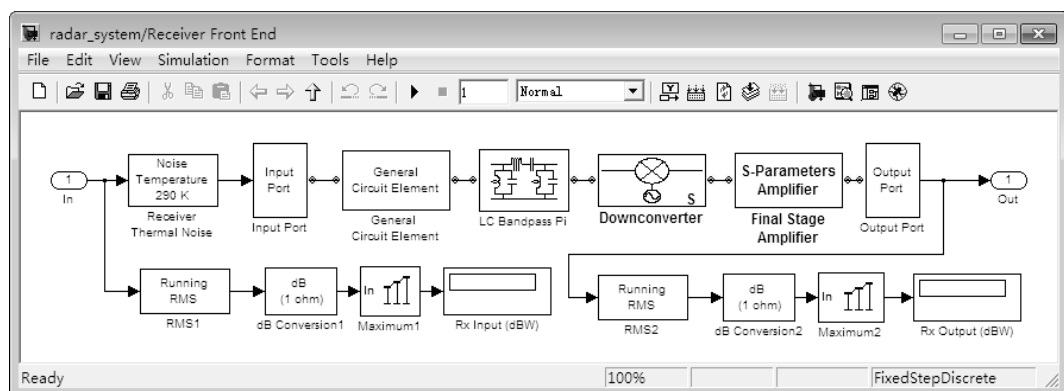


图 10-13 雷达射频接收端电路

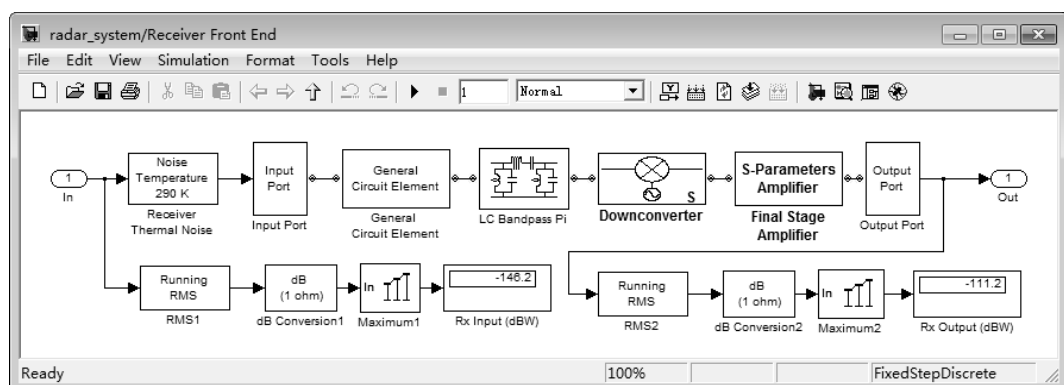


图 10-14 雷达接收端电路仿真结果

10.3 本例小结

本例以船舶雷达射频前端电路为对象,基于 RF 工具箱设计雷达射频前端电路的仿真系统,通过该例读者可以了解和掌握到基于 RF 工具箱建模、仿真的特点及流程,对于雷达射频前端电路结构和系统组成也将有所了解。



第 11 例 飞机供配电系统设计与仿真

飞机在现代社会中起着越来越重要的作用，它以其方便快捷等优点，已成为人们生活中不可或缺的工具。飞机供配电系统是保障飞机正常运作的基本系统之一。本例对其进行建模仿真。建模前首先介绍 Simscape 工具箱。

通过本例学习需了解和掌握以下几方面内容：

- 了解 Simscape 工具箱组成及特点。
- 进一步掌握基于 SimPowerSystems 工具箱进行电力系统设计方法。

11.1 Simscape 工具箱简介

11.1.1 Simscape 功能及特点

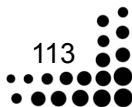
Simscape 是在 Simulink 基础上扩展的工具模块，用来搭建不同领域物理系统的模型，并进行仿真，例如由机械传动、机构、液压和电气元件组合而成的系统。Simscape 可以广泛应用于汽车业、航空业、国防和工业装备制造业。Simscape 同 SimMechanics、SimDriveline、SimHydraulics、SimElectronics 和 SimPowerSystems 一起，可以支持复杂的不同类型（多学科）物理系统混合建模和仿真。基于 Simscape 语言的 MATLAB 可以搭建物理元件、工具箱等。

Simscape 模型能够被转化成 C 代码（该过程需要使用 Real-Time Workshop）。C 代码可以用于 standalone 执行模式，并可集成到其他仿真环境下，例如 HIL 实时系统。

Simscape 能够用于搭建用户的电液阀、电气执行器、电阻、直流电机中的热量传递以及其他系统等。用户可以把 Simscape 模型和其他 MathWorks 物理建模产品联合使用从而实现多领域建模，例如电液联合，机电液一体化仿真等。

Simscape 具有以下几方面功能。

- （1）在统一环境中实现多种类型物理系统建模和仿真，包括机械、电气和液压系统。
- （2）使用基本物理建模单元构造模型，并提供了建模所需的模块工具箱和相关简单数学运算单元。
- （3）基于 Simscape 语言的 MATLAB，使用文本编辑搭建物理模型单元、域和工具箱等。
- （4）用户可自己指定参数和变量的单位，模块内部可自动实行单位转换和单位匹配。





(5) 具有连接不同类型物理系统的桥接模块。

(6) 具备扩展产品所建模型的全权仿真和受限编辑功能，单独运行仿真时无需 SimMechanics、SimDriveline、SimElectronics 和 SimHydraulics 的产品使用许可。

(7) 其最大的功能是能够进行多学科系统物理建模。

在 Simscape 的环境中，用户的建模过程如同装配真实的物理系统。Simscape 采用物理拓扑网络方式构建模型：每一个建模模块都对应一个实际的物理元器件，例如油泵、马达或者运算放大器；模块之间的连接线代表元件之间装配和能量传递关系。这种建模方式能直观地表现出物理系统的组成结构，而不是用晦涩的数学方程。Simscape 根据模型所表达的系统组成关系，自动构造出可以计算系统动态特性的数学方程。这些方程可同其他 Simulink 模型一起结合运算。

Simscape 模型中的 Sensor 模块用来测量机械量（力/力矩，速度），液压量（压力，流量）或电气量（电压，电流），测量输出的信号量可以输出给标准的 Simulink 模块处理。而 Source 模块能够将标准的 Simulink 信号转换成同等量值的上述物理信号。Sensor 和 Source 模块的使用将 Simulink 控制算法模型同 Simscape 物理网络拓扑模型有机地结合起来，可实现闭环控制算法开发。

11.1.2 Simscape 分类

Simscape 工具箱分类如图 11-1 所示，由图可知其可分为：基础子工具箱、传动系统子工具箱、电子系统子工具箱、液态系统子工具箱、机械子工具箱以及与其他系统连接用的子工具箱。下面分别予以介绍。

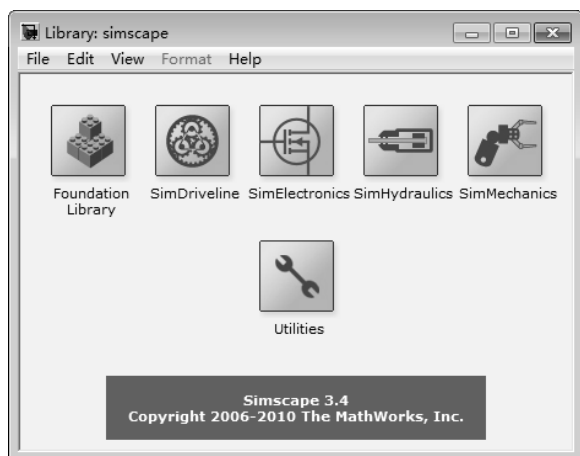


图 11-1 Simscape 分类

1. 基础子工具箱

该子工具箱提供一些基础的建模单元以便于建立多物理系统。使用基础元件工具箱



提供的建模单元，用户可以构建更多复杂的元件，扩展出更多种类的物理模型。在 Simulink 环境中，组合好的模型可以进一步封装形成可复用和共享的元件工具箱。

2. 传动系统与机械子工具箱

传动系统与机械子工具箱提供了一维平移/旋转机械的建模模块。机械系统元件的变量，将与系统中的液压和电气系统模型物理量同时解算。用于机械系统的 Sensor 和 Actuator 模块则可以连接由 SimMechanics 和 SimDriveline 创建的更精细的模型。

3. 电子系统子工具箱

电子系统子工具箱提供了代表电子元件和电路的电气模型。电气系统元件的变量，将与系统中的液压和机械系统模型物理量同时解算。用于电气系统的 Sensor 和 Actuator 模块则可以连接由 SimPowerSystem 创建的更精细的模型。

4. 液态系统子工具箱

液态系统子工具箱提供了液压系统的建模模块，可对基本的液压特性进行建模和计算；也可以进一步组合出更加复杂的液压元件。这些模块定义了各种物理过程中压力和流量的相互关系，如压缩性、流体惯性、摩擦损失、能量转换和固定节流口/可变节流口出流。通过输入流体属性数据，用户可以自己定义流体特性。使用 SimHydraulics 则可以构建更加精细的液压元件模型。此外还提供了热能搭建模块用于仿真热能效应。用户能够建模传导、对流以及散热等。使用热能源模块，用户能够确定温度或者热交换，使用热能传感器模块，用户能够测量热能值或者温度变化。

5. 其他常用模块子工具箱

该子工具箱提供一些其他常用模块以便于建立多物理系统。在 Simscape 中，用户可以直接操作物理量信号。信号和参数的单位直接在模块设定的对话框中输入，Simscape 在求解物理网络模型时将自动进行必要的单位转换。Physical Signal 模块工具箱支持对物理信号的数学操作，可以在物理模型网络中直接构建运算模型。Simscape 模块之间通过 Physical Signal ports 端口相互连接，这样各种类型的物理系统可以方便地集成。

用户还可在 Simscape 中使用传感器模块测量具有不同物理量纲的值，例如机械方面（力/力矩，速度），液压（压力，流速），或者电气（电压，电流）变量，并可把这些变量传递到标准 Simulink 模块中。

11.1.3 Simscape 数学方程及仿真流程

1. Simscape 建模用的数学方程

表征物理系统的数学工具为微分方程、代数方程，或者是两者的混合微分-代数方程。这三者的特点为：



- (1) 微分方程描述系统状态随时间的变化。
- (2) 代数方程描述系统变量间的约束限制，但是无系统变量的微分量。
- (3) 如果系统既有微分变量，变量间又存在约束条件，则系统需要通过微分-代数方程进行描述。

注意：Simscape 假设模拟的对象是一个微分-代数系统。

2. Simscape 仿真流程

Simscape 仿真流程如图 11-2 所示。

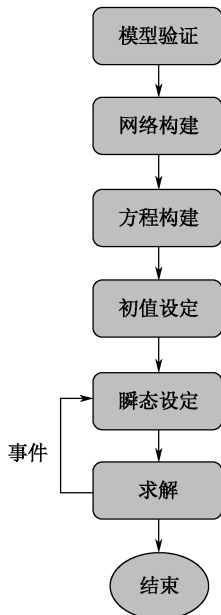


图 11-2 Simscape 仿真流程图

1) 模型验证

Simscape 解算器首先验证模型配置及参数设置是否正确。主要检查以下几项：

- 每一个独立的物理网络都必须有且仅有一个独立的解算器模块。
- 由于 Simscape 中规定模块及连线信号都必须有特定的物理单位，而 Simulink 中的模块则没有此限制。所以在两者模块间存在信号交互时，需注意物理量的统一。如果存在多种类型信号转换时，需检测此项。
- 如果模型中包含液态或气态模块，则每一个有液态或气态模块的独立物理网络中都必须包含一个“Hydraulic Fluid block”或“Gas Properties block”，这两个模块用于设定液态和气态特征参数。

2) 网络构建（拓扑构建）

验证完模型后，Simscape 解算器按照以下两个规则来搭建系统网络：

- 两个连接模块端口间截断变量值（across variable，例如电压、角度等）相同。
- 直通变量（through variable，如电流，力矩等）可产生多个分支，每个分支的数值



可能存在不同,但是对于一个分支点,流入直通变量总和与流出直通变量总和必须相等。

3) 方程构建

基于网络构建及初始参数设置,Simscape 解算器建立模型数学方程,由之前 Simscape 由微分-代数方程描述可知方程分为两部分:

- 动力学方程——描述模型中变量随时间变化的规律。
- 代数方程——描述变量间约束关系。

4) 初始状态计算

Simscape 解算器在初始 0 时刻计算一次初始状态。

Simscape 解算器计算初始状态时,将除在模块对话框设置初值外的变量都设为 0,并计算所有变量的值。

对于可自定义初始值的变量,需注意使定义的参数与系统其他变量初始值相统一。例如,两个并联的电容两端的初始电压需相等。

Simscape 解算器默认是不从稳态开始计算。如果勾选“Start simulation from steady state”选项,则模型从稳态开始仿真。

注意:如果勾选“Start simulation from steady state”选项仿真运行时失败,则可去掉该选项再次仿真。

5) 瞬态初始化

计算得到模型初始状态或发生一个事件后,Simscape 解算器进行瞬态初始化。瞬态初始化通过固定动态变量,计算代数变量及动态变量的微分,从而使初始状态一直并为数值求解提供基础。

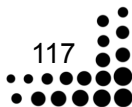
6) 求解

在得到初始状态及瞬时初始化之后,Simscape 利用特定的求解器求解下一时刻模型的状态。在下一时刻如果碰到一个事件(不连续状态、过零等),则返回瞬时初始化,如果没有,则继续解算,直至到达仿真结束时刻。

11.2 飞机供配电系统建模与仿真

飞机供电系统是现代飞机的一个重要组成部分,它的作用是向飞机上所有用电设备(如飞行控制系统、航空电子设备、火力控制和照明等)连续地提供满足规定技术性能的电能,保证用电设备的正常工作。飞机电子系统由供电系统和用电设备组成。供电系统是电能产生、控制、变换和输配系统,可分为电源系统和配电系统。

飞机电源系统由主电源、辅助电源、应急电源、二次电源和地面供电接口等组成。主电源是直接由航空发动机驱动或通过恒速传动装置传动的发电机及其控制保护器组成。辅助电源是蓄电池或者自吸发动机或涡轮驱动的发电机。应急电源是一个独立的电源,主电源都发生故障时,向飞机重要用电设备供电。二次电源是变换了主电源的电压、频率和电流种类,给专门用电器供电的电源类型。现代飞机主电源有低压直流电源、恒速恒频交流电源等。飞机上装有两种或两种以上主电源的电源称为混合电源。





配电系统式连接电源与用电设备的重要环节，是实现余度容错和不间断供电的基础，由导线或电缆、配电装置、保护装置及检测仪表等构成。常规式配电系统功率线全部引入座舱内的配电中心或中心配电装置，二级配电中心或电气负载从中心配电装置获得电能，由断路器提供馈电线过载保护。

11.2.1 飞机供配电系统电路设计

飞机供配电系统电路如图 11-3 至图 11-10 所示。图 11-3 给出一个通用的飞机供配电电路，其中交流电的频率是可变的并取决于发动机速度。系统由六大部分组成。

第一部分是发电机驱动，该部分由一个信号发生器进行模拟，该驱动器给出发电机规划速度。

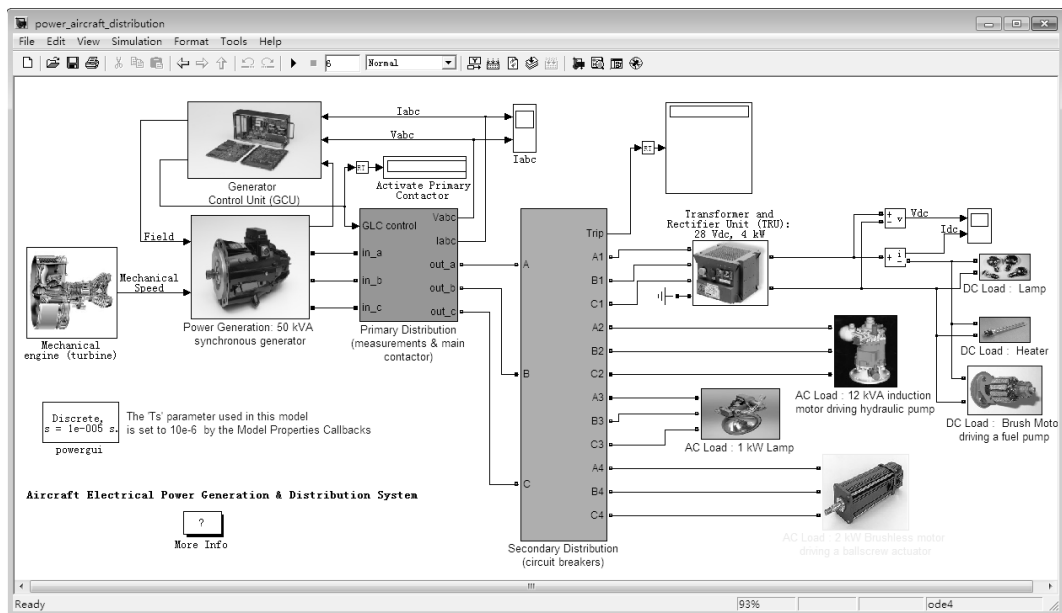


图 11-3 飞机供配电系统框图

第二部分是发电装置。发电装置由一个简化的同步电机与电压控制单元构成，同步电机产生交流电。电压控制单元将电压控制在 200V。

第三部分是一级配电系统。除了三个电压及电流敏感器外，还包括由电压控制单元输出的三相接触开关。另外，为避免数值奇异，需要加上适当的阻抗。

第四部分是二级配电系统。它包含四个断路器及相应的可调电流跳闸。

第五部分是交流电负载。它包含一个 4kW 变压器、一个整流机组、一个 12kW 的感应电机和一个 3 码的无刷直流电机驱动。

第六部分是直流电负载。包含两个阻性负载（加热与照明）和一个 300W 的直流电机（用于驱动燃料泵）。

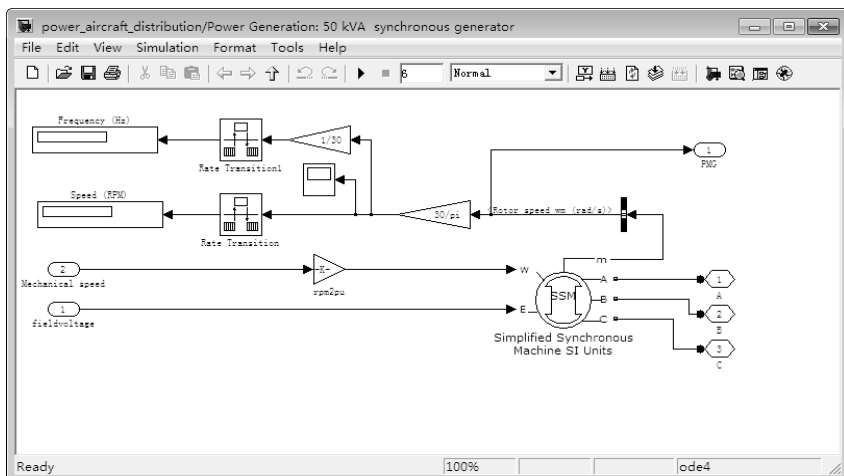


图 11-4 同步发电机

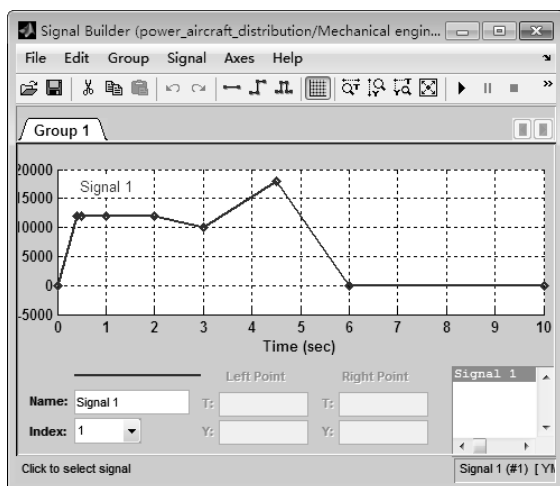


图 11-5 发电机驱动信号

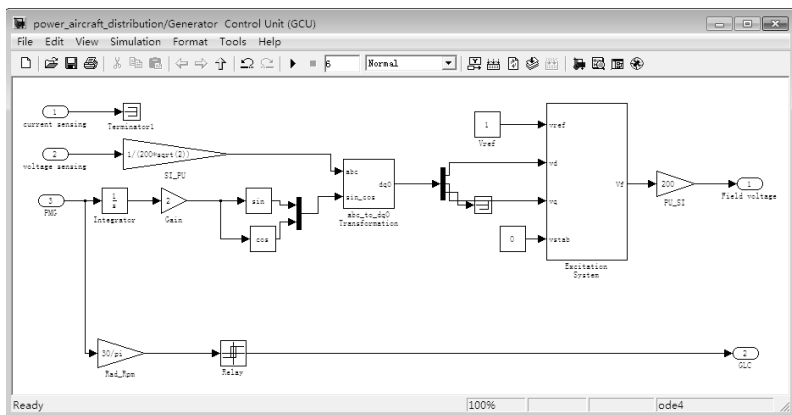


图 11-6 电压控制单元

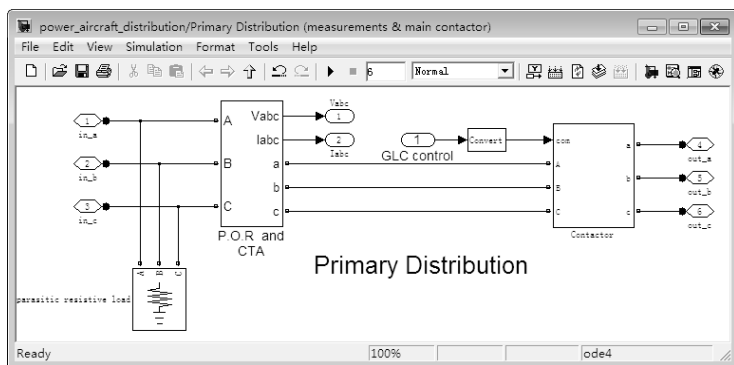


图 11-7 一级配电系统

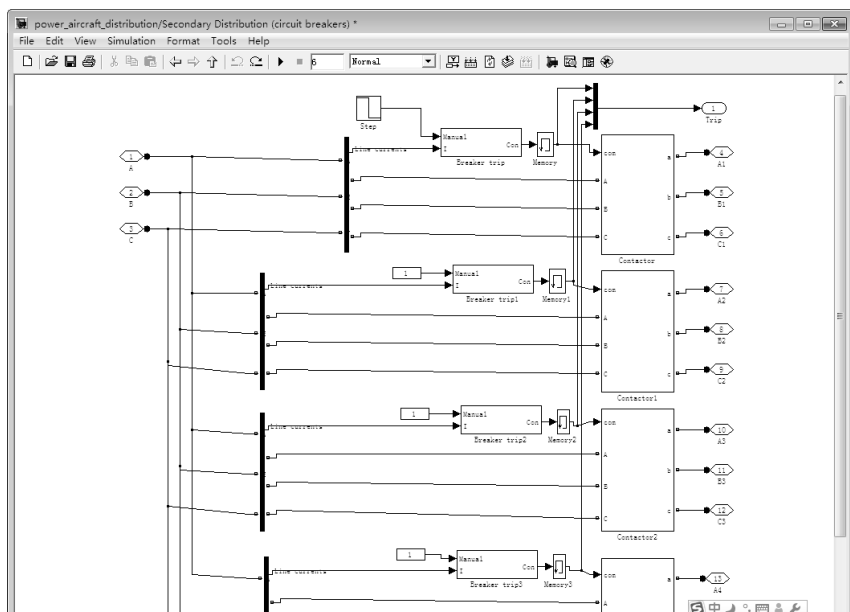


图 11-8 二级配电系统

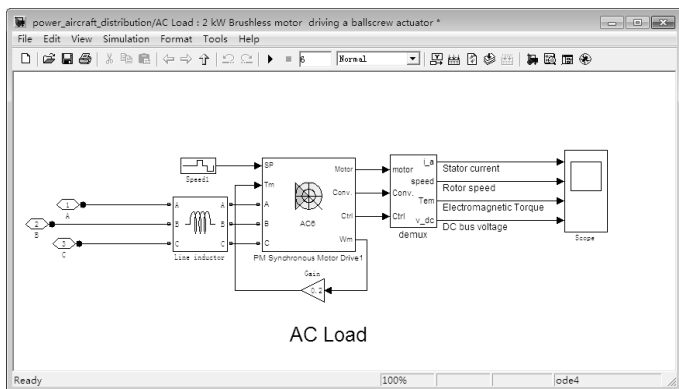


图 11-9 交流负载

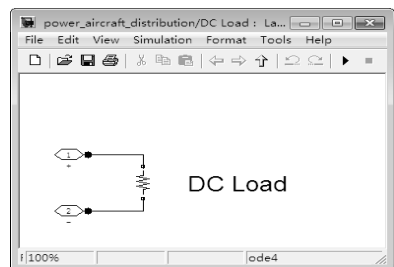


图 11-10 直流负载



11.2.2 仿真结果

仿真结果如图 11-11 和图 11-12 所示。

从第 0 秒开始，发电机在 0.4 秒内从 0 转每分钟加速度至 12000 转每分钟。

在第 0.3 秒，发电机速度升至 9000 转每分钟。电压控制单元打开主分配接触开关，开始为飞机提供交流电。阻抗负载已入电网。直流电母线电压升至 28V。感应电机和直流电机转速加速至常速。

在第 1.4 秒，无刷直流电机加速至指定的 500 转每分钟。

在第 1.9 秒，无刷直流电机加速至指定的-500 转每分钟。这个时候主电流开始下降，因为这个时候的电机具有发电机功能。

在第 2 秒，发电机在 1 秒内转速由 12000 转每分钟降至 10000 转每分钟。通过观测感应电机的转速，可以发现其随着发电机频率的下降而下降。

在第 3 秒，发电机在 1.5 秒内由转速 10000 转每分钟加速至 18000 转每分钟。

在第 3.5 秒，整流机组被断开，这使得主电流下降。

在第 4.5 秒，发电机在 1.5 秒内转速由 18000 转每分钟降至 0。

在第 5.26 秒，发电机转速已降至 8900 转每分钟，此时电压控制机构将主配电系统断开。

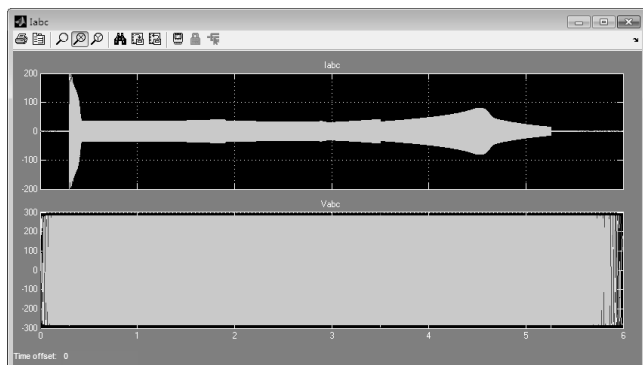


图 11-11 发电机功率

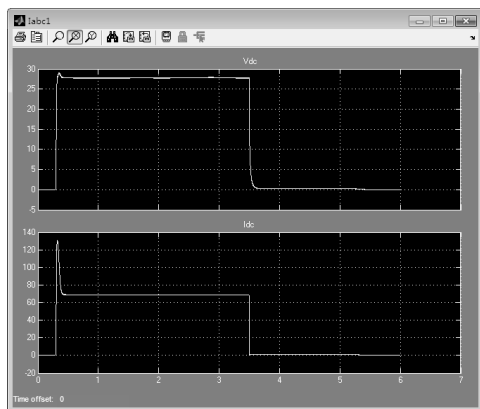
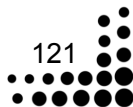


图 11-12 二次分配交流负载电压





11.3 本例小结

本例以飞机为对象，基于 SimPowerSystems 设计飞机供配电仿真系统，通过该例读者可以进一步了解和掌握到基于 SimPowerSystems 建模、仿真的特点及流程，对于飞机供配电系统组成也将有所了解。



第 12 例 重力场卫星加速度计 读取电路设计



重力场卫星用于地球重力场模型测量，在导航及地球观测中能起到重要的作用，是我国目前航天领域内研究的重点之一。加速度计是重力场卫星核心部件，这里基于 SimElectronics 工具箱设计加速度计读取电路。

通过本例需了解和掌握以下几个方面内容：

- 了解和掌握 SimElectronics 工具箱特点。
- 了解加速度计读取电路结构。
- 掌握基于 SimElectronics 工具箱进行电路设计流程。

12.1 SimElectronics 工具箱简介

12.1.1 SimElectronics 工具箱特点

1. SimElectronics 工具箱特点

SimElectronics 是 Simulink 多物理仿真中的一部分。通过 SimElectronics 及其他物理仿真工具箱可以完成对象系统级多物理仿真任务。

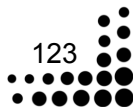
SimElectronics 里面的模块不仅可以和 Simscape 中其他模块直接相连，还可以通过转接模块与 Simulink 里的模块进行连接，从而使得建模更加迅速快捷，仿真功能更加强大。

SimElectronics 是 Simscape 的扩展，它的运行需要其他三个部件：

- MATLAB。
- Simulink。
- Simscape。

2. 打开 SimElectronics 工具箱的方式

(1) 通过单击 MATLAB 工具栏中 Simulink 按钮或者在 MATLAB 中输入“Simulink”可打开 Simulink 工具箱浏览器，找到 SimElectronics，如图 12-1 所示。SimElectronics 是 Simscape 中的一部分。Simscape 中其他还有 SimDriveline、SimHydraulics、SimMechanics 等工具箱，这些工具箱分别对传动系统、柔性系统和机械系统进行建模仿真。由于它们





都是 Simscape 中的一部分，因此它们之间可以进行无缝连接，从而建立多物理系统仿真。

(2) 通过在 MATLAB 中输入命令打开。

在 MATLAB 命令行中输入“elec_lib”可打开 SimElectronics 工具箱，输入“simscape”可打开 Simscape 工具箱。

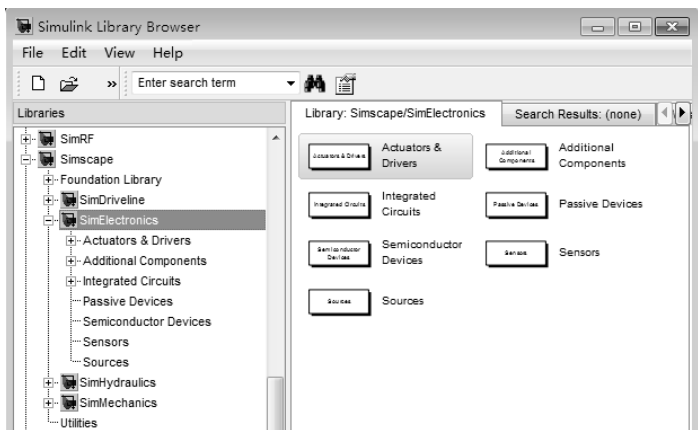


图 12-1 SimElectronics 工具箱

12.1.2 SimElectronics 工具箱分类介绍

由图 12-1 右侧可知 SimElectronics 工具箱共有七个子工具箱，下面分别对其予以介绍。

1. 执行机构与驱动子工具箱

执行机构与驱动子工具箱如图 12-2 所示。由图可知其包含三个更小的子系统，其中“Drivers”为驱动电路，“Rotational Actuators”为旋转执行机构，“Translational Actuators”为直线执行机构。

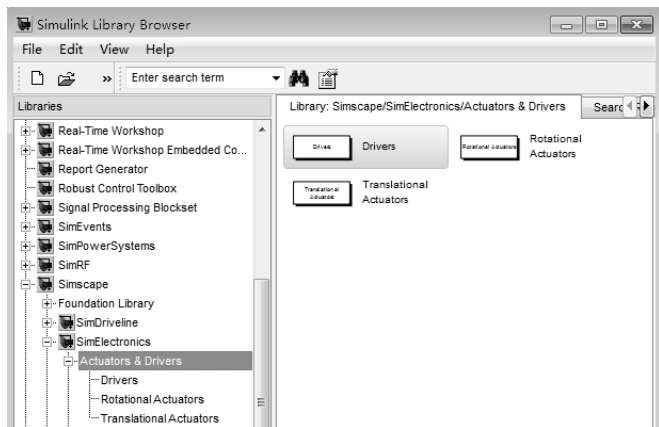


图 12-2 执行机构与驱动子工具箱

2. 集成电路子工具箱

该子工具箱如图 12-3 所示，由图可知其包含以下几个模块：

- Band-Limited Op-Amp有限带宽运算放大器。
- Comparator比较器。
- Finite-Gain Op-Amp有限增益运算放大器。
- Multiplier集成运算放大器。
- Timer计时器。

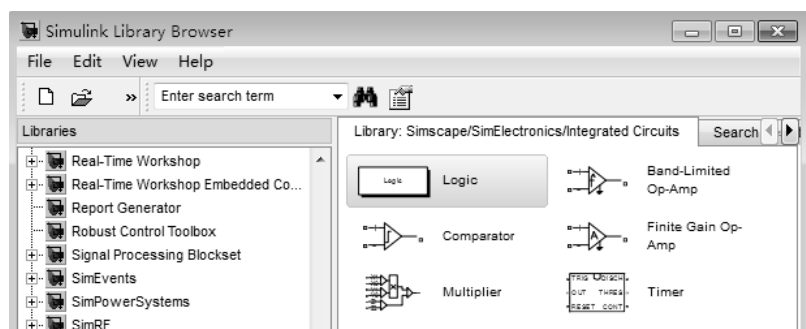


图 12-3 集成电路工具箱

3. 无源模块子工具箱

该子工具箱如图 12-4 所示，由图可知其包含以下几个模块：

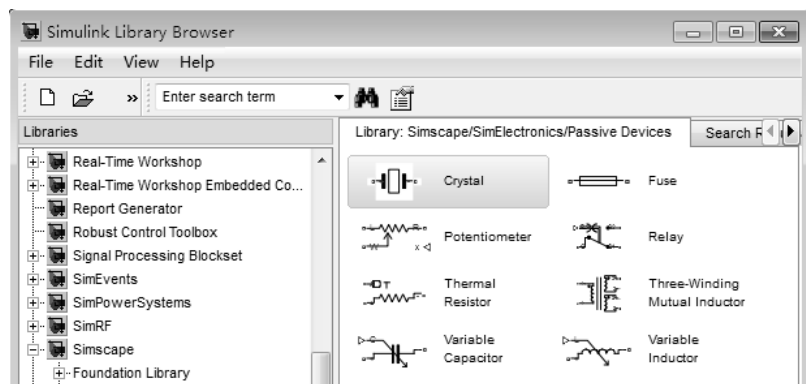


图 12-4 无源模块子工具箱

- Crystal晶振。
- Fuse电容丝。
- Potentiometer电位器。
- Relay中继器。
- Thermal Resistor热阻器。
- Three-Winding Mutual Inductor三绕组互感器。
- Variable Capacitor线性时变电容。



- Variable Inductor 线性时变电感。

4. 半导体模块子工具箱

该子工具箱如图 12-5 所示，由图可知其包含以下几个模块：

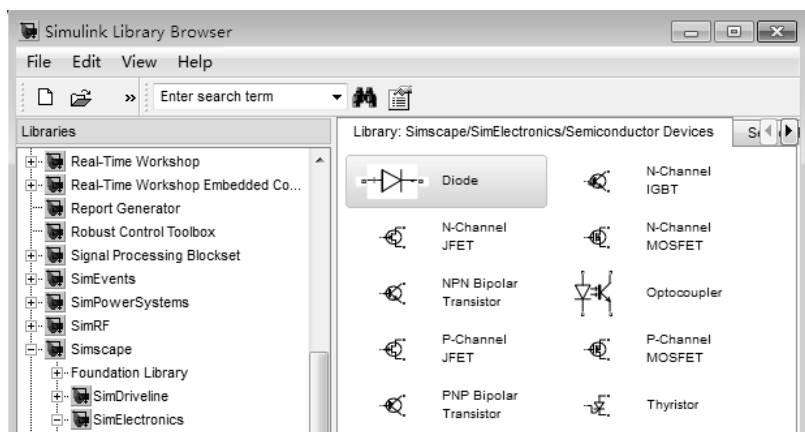


图 12-5 半导体模块子工具箱

- Diode 二极管模型。
- N-Channel IGBT N 沟道绝缘栅双极型晶体管模型。
- N-Channel JFET N 沟道场三级晶体管模型。
- N-Channel MOSFET N 沟道场效应晶体管模型。
- NPN Bipolar Transistor NPN 双极型晶体管。
- Optocoupler 光耦模型。
- P-Channel JFET P 沟道场三级晶体管模型。
- P-Channel MOSFET P 沟道场效应晶体管模型。
- PNP Bipolar Transistor PNP 双极晶体管。
- Thyristor 晶闸管模型。

5. 电源子工具箱

该子工具箱如图 12-6 所示，由图可知其包含以下几个模块：

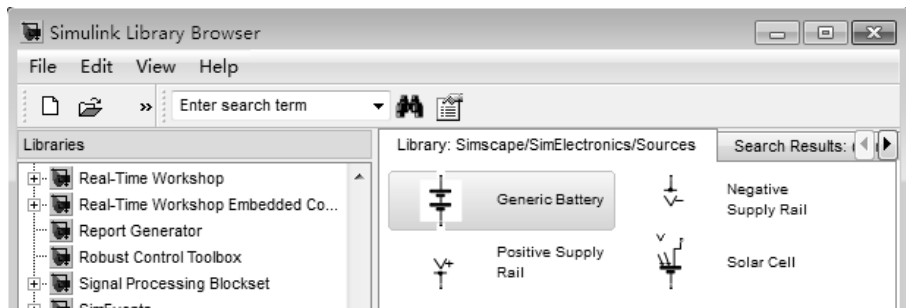


图 12-6 电源子工具箱

- Generic Battery 通用电池模型。
- Negative Supply Rail理想负电源模型。
- Positive Supply Rail理想正电源模型。
- Solar Cell单太阳电池模型。

12.2 重力场卫星加速度计敏感电路设计与仿真

重力场卫星用于测量地球重力场模型，地球重力场在地球物理学、海洋学和空间技术中占有特别重要的地位。它直接反映地球内部的密度分布。从地幔产生的长波信号，到大陆岩石圈和海底地壳的局部特征等，都反映在地球重力场中。用一组重力位系数来表示相应的地球重力场，称为地球重力场模型。它的作用可简单归纳为以下几点。

(1) 卫星大地测量定位的精度取决于卫星定轨的精度，而全球重力场模型是精密定轨的基础。

(2) 通过地球重力场模型及对地球外部重力场的分析，可为地球物理学和地质学提供地球内部结构和状态的信息。

(3) 地球重力场模型可精确确定地球的扁率。

(4) 各国的区域性坐标系与全球坐标系的精确转换，需要区域性大地水准面资料，而大地水准面属地球重力场的一个等位面。

(5) 大地测量观测是在地球重力场内进行的，数据的处理和归算要知道地球重力场。

(6) 人造卫星、洲际导弹轨道的摄动与地球外部重力场密切相关。

(7) 重力勘探是重力学原理在勘探地下资源方面的应用，根据局部重力场变化规律可以反推矿藏位置和范围。

12.2.1 重力场卫星读取电路设计

重力场卫星中的一个最关键的部件是加速度计，而加速度计读取电路又直接决定了读取精度的高低。典型的加速度计读取电路如图 12-7 所示。

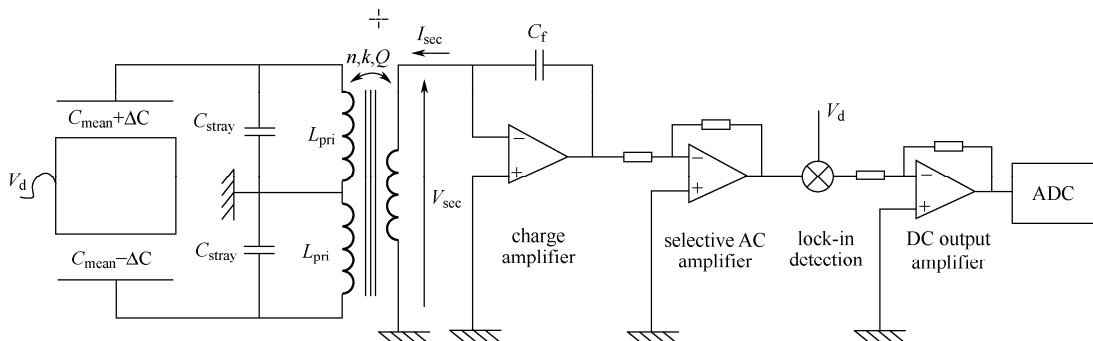


图 12-7 重力场卫星加速度计读取电路原理图



从原理图中可以看出，电路可分为六部分：差分电桥式电容传感器电路、电流放大器、选频器、相敏检波器、直流放大器和模数转换器。

加速度计利用电容传感器来测量加速度计中质量块相对于容器壁的相对距离，其中容器壁上装有电极，电极与质量块之间形成电容，当质量块位置发生变化时，电容也发生变化。通过差分电桥可以将这一电容变化检测出来，从而可以得到质量块位置变化。

经过差分电桥检测出来的是微小的交流电压信号，为得到准确的位置信息，需要对该信号进一步处理。电流放大器是将该信号进一步放大；选频器是选取所需频段的信号，而将其他信号过滤掉；相敏检波器是将选取指定频率的信号，并将交流信号转为直流信号；直流放大器进一步将直流信号放大；数模转换器将模拟信号转为数字信号以便后续处理。

这里设计的读取电路包含其中的差分电桥式电容传感器电路、电流放大器。

12.2.2 仿真结果及分析

基于 MATLAB 电子工具箱设计的重力场卫星加速度计读取电路模型如图 12-9 至图 12-11 所示。其中图 12-9 为整个电路系统图，图 12-10 为差分电桥式电容传感器电路，图 12-11 为可变电容器模块，图 12-12 为电流放大器电路。

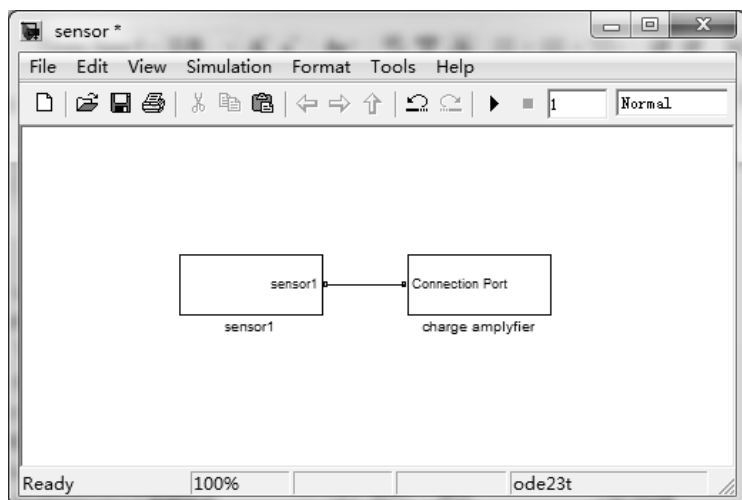


图 12-9 重力场卫星加速度计读取电路框图

图 12-10 中电容器为可变电容器，其特点是电容大小随着加速度计内部质量块位置的变化而变化，为仿真其性能，基于 Simscape 语言设计一个新的可变电容器模块，如图 12-11 所示。为基于 Simscape 语言自定义的模型，定义的语言如下所示。关于 Simscape 语言将在下一部分详细介绍。

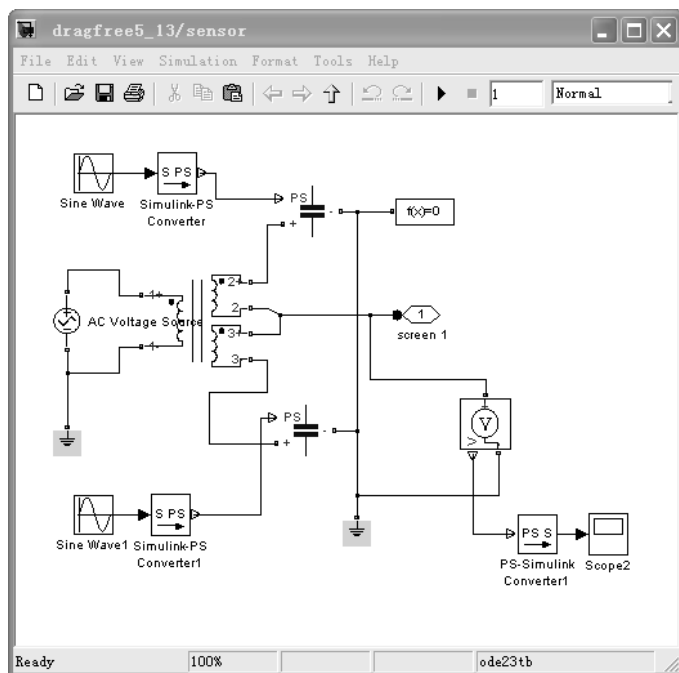


图 12-10 差分电桥式电容传感器电路



图 12-11 可变电容器模块

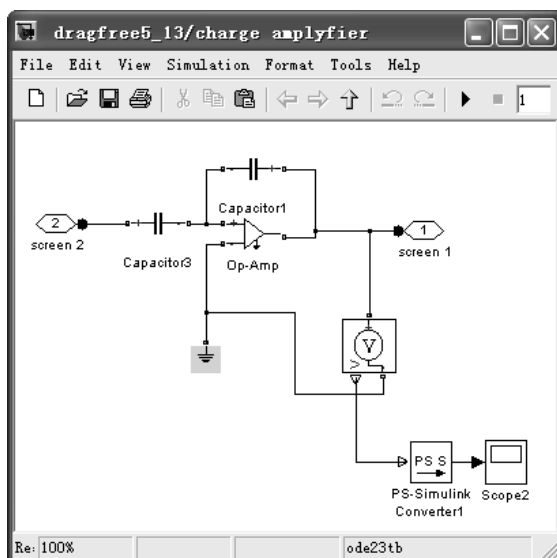


图 12-12 电流放大器电路

```
component IdealCapacitor
inputs
    cc = { 0, 'F' }; % +:top
end
nodes
    p = foundation.electrical.electrical; % +:top
```



```
n = foundation.electrical.electrical; % -:bottom
end
parameters
    C = { 1, 'F' };
    V0 = { 0, 'V' };
end
variables
    i = { 0, 'A' };
    v = { 0, 'V' };
end
function setup
    if C <= 0
        error( 'Capacitance must be greater than zero' )
    end
    through( i, p.i, n.i );
    across( v, p.v, n.v );
    v = V0;
end
equations
    i == (C+cc)*v.der;
end
end
```

如图 12-13 至图 12-14 所示为仿真结果。其中图 12-13 为差分电桥式电容传感器电路输出电压，从图中可以看出，输出电压为一个受正弦波信号调制的基频曲线。其中基频频率较高，而正弦波信号为需要检测的质量块运动的信号。读取电路就是要将该正弦信号辨识出来。

图 12-14 是经过电流放大器放大后的电压信号，由图可看出，信号经过放大器增强了很多，从而有利于进一步处理。

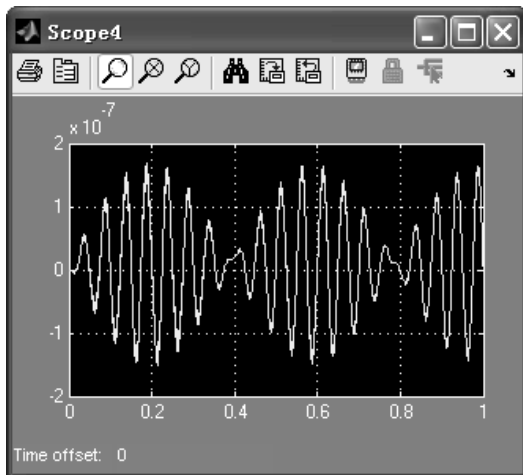


图 12-13 差分电桥式电容传感器电路输出电压

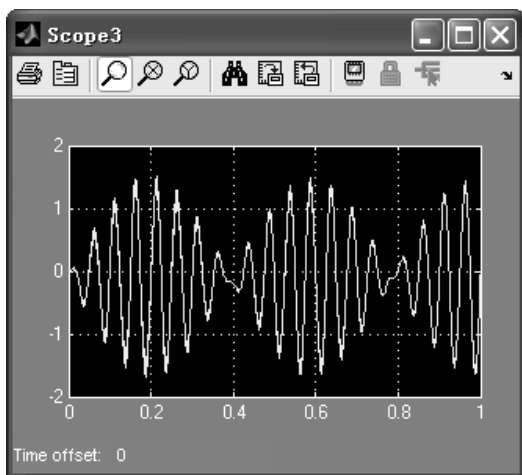


图 12-14 经电流放大器放大后电压

12.3 本例小结

本例以重力场卫星加速度计为对象，基于 SimElectronics 建立加速度计读取电路仿真系统，通过该例读者可以了解和掌握到基于 SimElectronics 建模、仿真的特点及流程，对于重力场卫星加速度计读取电路组成也将有所了解。

第四部分 结构工程仿真实例



引言——SimMechanics 工具箱简介（上）

一、SimMechanics 工具箱功能概述

SimMechanics 工具箱是 Simscape 工具箱下面的一个子工具箱，通过一系列工具模块完成对机械系统的建模与仿真。它与 Simulink 模块之间的连接通过敏感器及执行机构模块完成。

SimMechanics 工具箱主要按照以下几个步骤完成对机械系统建模与仿真。

- (1) 制定物体惯性属性、自由度、约束以及限定与某坐标系的观测变量和执行变量。
- (2) 制定敏感器用于记录运动与力，以及建立初始化所需要的执行机构及初始力。
- (3) 开始仿真。
- (4) 在仿真的同时，利用 SimMechanics 工具箱的可视化窗口同时观测仿真过程。

二、SimMechanics 工具箱组成

1. 打开 SimMechanics 工具箱的方式

- (1) 在命令行中输入命令 `mechlib`，打开后如图 D1 所示。

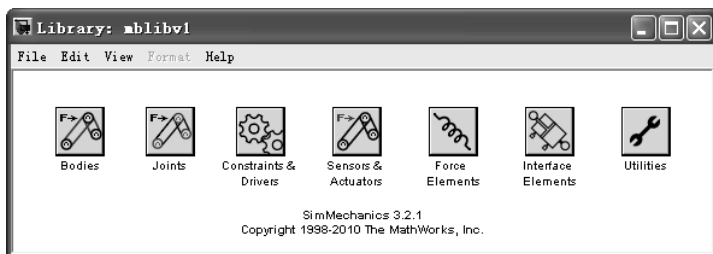


图 D1 SimMechanics 工具箱命令行打开方式



(2) 打开 Simulink，在左侧列表中找到 SimMechanics，单击打开，打开后如图 D2 所示。

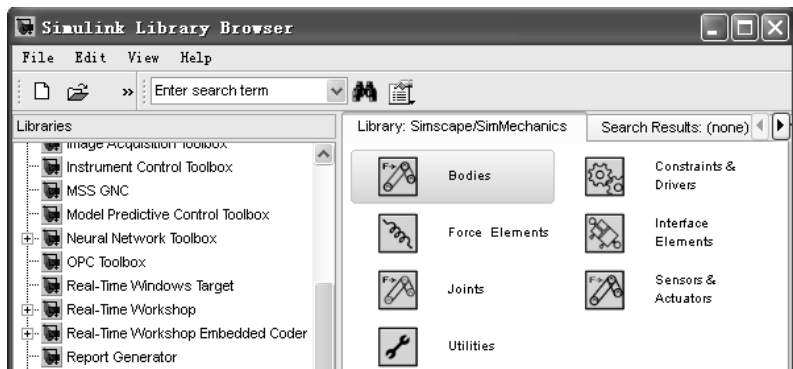


图 D2 SimMechanics 工具箱列表打开方式

2. SimMechanics 工具箱中的子工具箱

(1) Bodies——刚体子工具箱。该子工具箱如图 D3 所示，共包含四个模块。第一个模块是刚体模块 (Bodies)，用于定义物体的质量、转动惯量、位置、指向以及关联的坐标系等；第二个模块是地模块 (Ground)，用于定义不可移动的地面上的点；第三个模块和第四个模块是环境模块，用于配置 SimMechanics 工具箱的参数。

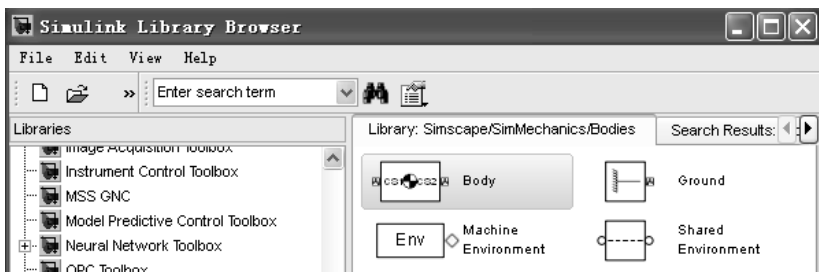


图 D3 刚体子工具箱

(2) Joints——连接子工具箱。该子工具箱如图 D4 所示，用于表示刚体间自由度的关系。模块分为三类：第一类是装配连接模块，这一类模块能够限制相互连接刚体的自由度，可通过敏感器和执行机构对其进行观测和驱动；第二类是非装配连接模块，这类模块并不限制自由度，用于机械连接回路的两端，不可观测和驱动；第三类是无质量连接模块，这类模块用于描述存在一定距离的物体间关系，同样不可观测和驱动。

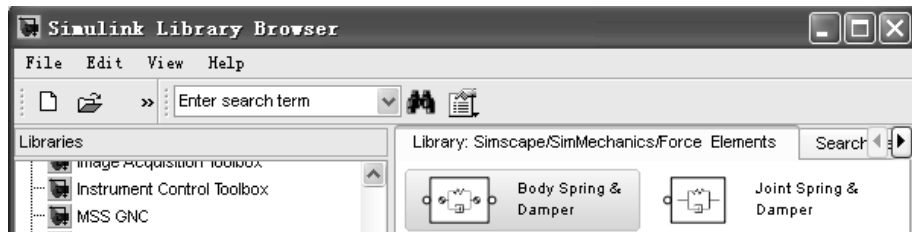
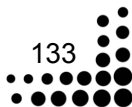


图 D4 连接子工具箱





(3) Constraints&Drivers——约束及传动子工具箱。该子工具箱如图 D5 所示，表示物体间的约束及传动关系。其中包含角传动、距离传动、齿轮传动、线性传动、平行传动、点传动和速度传动等模块。

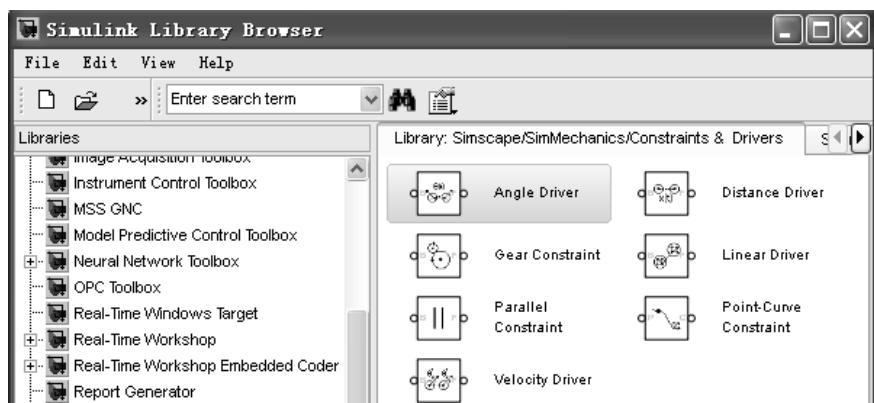


图 D5 约束及传动子工具箱

(4) Sensors&Actuators——敏感与执行子工具箱。该子工具箱如图 D6 所示，用于观测物体状态及驱动物体运动。其中包含物体执行机构、物体敏感器、约束及传动敏感器、传动执行机构、连接执行机构、连接初始状态、连接敏感器和变质量执行机构等。这些模块是连接 Simulink 其他模块的主要途径，从图中可以看出，模块前后端口的形状不一样，有一侧用于连接 Simulink 模块，另一侧用于连接 SimMechanics 模块。

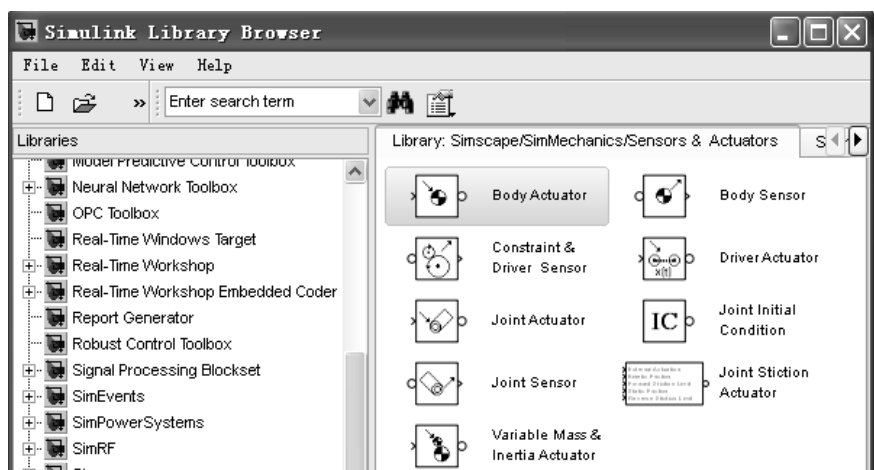


图 D6 敏感与执行子工具箱

(5) Force Elements——力子工具箱。用于生成物体间耦合力及力矩。共有两个模块：物体间弹簧-阻尼模块和连接间弹簧-阻尼模块。

(6) Interface Elements——接口子工具箱。该子工具箱如图 D7 所示，用于连接 SimMechanics 模块和 Simscape 模块。共有连接平动和转动两个模块。

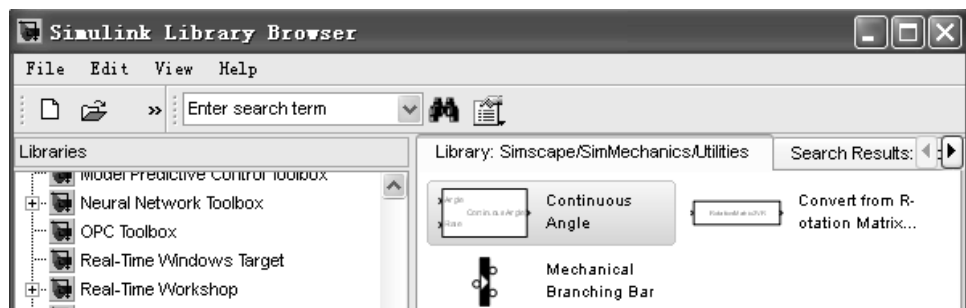


图 D7 接口子工具箱

(7) Utilities——辅助子工具箱。该子工具箱如图 D8 所示，用于辅助基于 SimMechanics 模块建立机械对象。共有三个模块：第一个模块用于产生连续的角度信号；第二个模块用于将 3×3 矩阵转换成四元数，从而便于在三维图形中显示；第三个模块用于将机械多维信号分开。

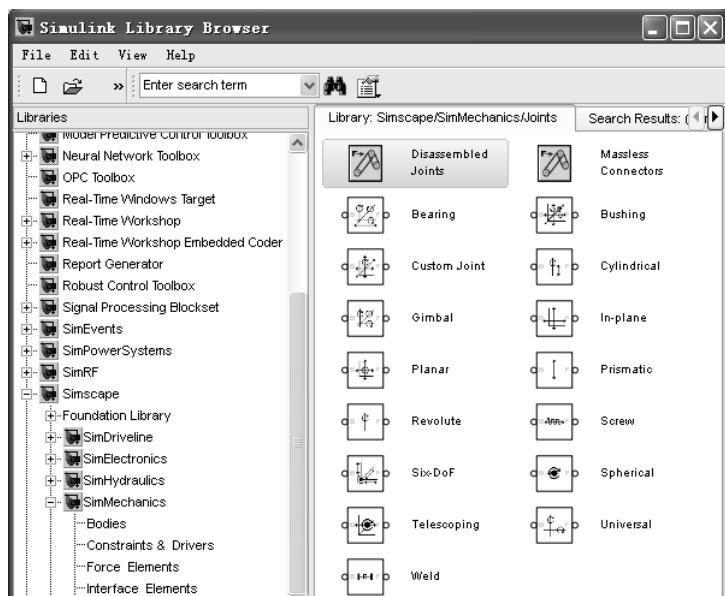


图 D8 辅助子工具箱

三、SimMechanics 工具箱建模

基于 SimMechanics 工具箱建模分为以下几个步骤。

1. 建立刚体和地模型

任何一个机械系统最重要的部分都是组成的刚体。基于 SimMechanics 工具箱的刚体模型与一般的刚体模型存在不同，这里将质量特性与自由度分开。SimMechanics 的刚体模型只包含质量特性，而自由度需另外建立模型进行定义。



首先建立地模型。地指一个理想的固定不变的参考点。地模型通过自由度组件与刚体模型连接在一起，自由度组件定义了刚体相对于地的自由度。如果要定义机械对象其他部件相对环境的相对位置，则需要定义另外的地模型。

注意：每个 SimMechanics 模型中至少有一个地模型。通过“Shared Environment”模块可使不同刚体共享同一环境。同时地模型必须连接到“Machine Environment”模块。

其次建立刚体模型。刚体是实际固体的理想化模型，即在受力后其大小、形状和内部各点相对位置都保持不变的物体。描述刚体的惯性特征为质量和转动惯量，而刚体运动分为平动和转动，平动对应位置，转动对应角度。

按照以下几个步骤建立刚体模块。

- (1) 从 SimMechanics 刚体库中拖拽一个刚体模块至模型窗口中。
- (2) 打开刚体模块属性对话框。
- (3) 输入刚体质量。
- (4) 选择质量单位。
- (5) 输入刚体转动惯量。
- (6) 输入刚体初始位置。
- (7) 输入刚体初始姿态角。

2. 建立自由度

SimMechanics 连接件描述一个刚体相对于另一个刚体的自由度。基准刚体可以是一个运动的刚体，或是地。在 SimMechanics 中将自由度与质量特性分开，所以这里的连接件模型不涉及质量。

注意：SimMechanics 中只有连接件一端连接地，另一端连接刚体，才是描述刚体相对于惯性坐标系的运动。

对自由度进行建模，需要了解以下几个概念。

1) 运动副

两构件直接接触并能产生相对运动的活动连接称为运动副。两个构件上参与接触形成运动副的点、线、面等元素被称为运动副元素。运动副限制了两构件之间的某些相对运动。表 D1 列出了 SimMechanics 中运动副分类。

表 D1 SimMechanics 中运动副分类

运动副类型	符 号	自 由 度
移动铰链 (Prismatic)	P	一维平动
转动 (Revolute)	R	一维转动
球接 (Spherical)	S	三维球接
焊接 (Weld)	W	无自由度

2) 连接类型

SimMechanics 中连接类型主要有：运动副连接、复合运动副连接、无质量连接、装

配连接等。其中运动副连接指采用基本运动副连接中的一种进行连接；复合运动副连接指采用多个基本运动副进行连接；无质量连接指多个连接之间存在固定的距离；装配连接指连接的两个刚体按照一定的装配关系进行连接。

3) 连接轴

连接模块定义了一个或多个连接轴。例如移动铰链定义了一个平动轴，转动连接定义了一个转动轴，球接定义了旋转轴的支点，而平面连接定义了两个平动连接轴。

连接轴有连接轴方向、连接轴阶数和连接轴空间等三个特征。连接轴方向指连接轴的指向。连接轴阶数指多个运动副共同定义的连接轴中沿每个轴或支点跟随刚体移动的顺序。连接轴空间指连接轴移动所经过的空间，例如转动连接中连接轴空间为一条直线，平面连接中连接轴空间为一个平面。

4) 连接方向

连接方向指相对运动的方向性，定义了正负。例如移动铰链是沿着直线运动，定义一个方向为正，相反的方向为负。转动连接是在一个平面内转动，定义往一个方向旋转为正，相反的为负。

基于 SimMechanics 建立连接的过程为：

- (1) 从 SimMechanics 连接库中选择连接模块并拖拽进模型窗口。
- (2) 将一端刚体模块与连接模块连接起来。
- (3) 将剩下的另一端或多端模块与连接模块连接起来。
- (4) 定义连接轴方向。
- (5) 如果要添加敏感器或执行机构，则创建新的端口来连接。

图 D9 给出了连接的一个实例，由图可看出，地模块（Ground）通过转动连接模块（Revolute）与刚体模块（Body）连接，而两个刚体模块通过无质量连接模块（Revolute-Revolute）连接。

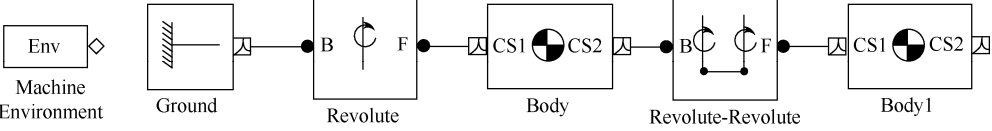


图 D9 连接实例

3. 建立约束与传动

SimMechanics 中约束与传动库提供了一系列模块来对约束和传动进行定义两个刚体之间的相对运动。图 D10 给出约束实例，图中通过平行约束使两个刚体在运动时保持平行。

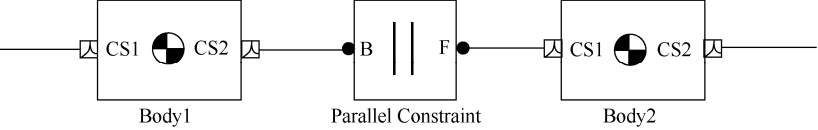


图 D10 约束实例



根据约束是否变化, 可将约束分为随时间不发生变化的约束和随时间发生变化的约束。而随时间变化的约束根据是否与速度量相关分为只与相对位置相关的约束和与相对速度相关的约束, 通过公式给出三类约束的区别: 式 (D1) 描述了与时间无关的约束, 式 (D2) 描述了只与相对位置相关的约束, 式 (D3) 描述了相对速度相关的约束。

$$f(x_B, x_F) = 0 \quad (D1)$$

$$f(x_B, x_F; t) = 0 \quad (D2)$$

$$f(x_B, \dot{x}_B, x_F, \dot{x}_F; t) = 0 \quad (D3)$$

如图 D11 所示约束实例, 由图中看出该约束为一个齿轮约束, 两个刚体在运动过程中需要保证外相切的约束关系, 这是一个与时间无关的约束。

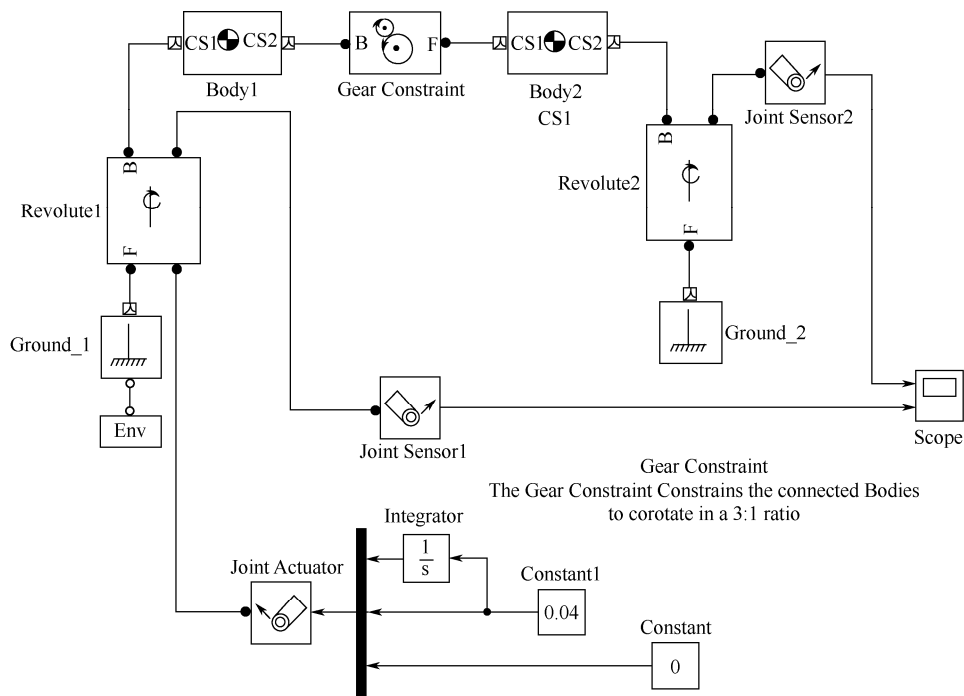


图 D11 约束实例

4. 切断回路

SimMechanics 中通过闭环来完成一个回路。闭环指从一点出发, 通过一条路径返回到起点。如果包含以下三部分, 则闭环有效:

- 至少一个连接模块。
- 不超过一个装配模块。
- 不超过一个约束或传动模块。

切断回路需遵循以下三个准则:

- 如果一个回路中包含约束、驱动或装配连接, 则解算器在这三类模块中任意一处



切断，事先的切断无效。

- 如果一个回路中不包含约束、驱动或装配连接，则解算器优先选择定义切断的地方进行切断。
- 如果一个回路中不包含约束、驱动或装配连接，也没有事先定义切断的地方，则解算器在自由度最多的地方进行切断。

5. 施加运动与力

SimMechanics 中传感器与执行机构库中提供了一系列执行机构模块用于添加随时间变化的力或运动于刚体、连接件或传动部件上，也可以改变质量或转动惯量。

注意：不要将执行机构与地连接，这是因为地是不可移动的。

以驱动刚体运动为例介绍添加执行机构模块的过程，驱动刚体运动按照以下几个步骤。

- (1) 如果目前没有连接端口，则建立一个体坐标系。
- (2) 拖拽一个刚体执行机构模块进模型窗口，并将其与体坐标系模块连接。
- (3) 打开执行机构模块，选择添加力或力矩于刚体上，并选择合适的单位。
- (4) 选择添加的力或力矩所位于的坐标系。
- (5) 通过 Simulink 建立力或力矩模型。
- (6) 将力或力矩模型连接到执行机构模块。

6. 观测运动与力

SimMechanics 中传感器与执行机构库中提供了一系列传感器模块用于观测刚体运动，连接部分的力、力矩、运动和约束的反力及力矩。

以观测刚体运动为例介绍添加观测模块的过程，观测刚体运动按照以下几个步骤。

(1) 如果刚体模块没有定义体坐标系，则定义一个。

(2) 拖拽一个传感器模块进模型窗口。

(3) 将传感器与刚体坐标系模块连接。

(4) 打开传感器模块，定义需要观测的量以及在哪一个坐标系中显示，如图 D12 所示传感器模块参数配置图，从图中可以看出“With respect to CS”定义了在哪一个坐标系，下面的几个选项卡是定义需要观测的量。

(5) 如果选择了多个观测变量，则需要勾选“Output selected parameters as one signal”。

(6) 将观测模块输出端口连接到 Simulink 输出模块。

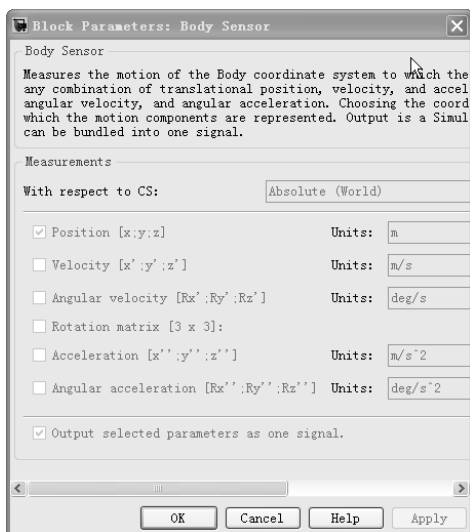


图 D12 刚体传感器参数配置



7. 添加内部力

内部力是由自身运动引起的作用于机械上的力。与需要通过 Simulink 信号添加外力不同，内部力由自身运动直接产生。

力元素库中已经提供了建立好的内部力模块，用户可以利用这些模块来建立内部力模型，也可以自己定义内部力模型。



第 13 例 车载stewart平台建模与仿真

本例在介绍 SimMechanics 工具箱仿真部分的基础上,以 stewart 平台为对象,介绍建立机械对象并仿真的过程。通过本例学习,需要掌握以下两点:

- 基于 SimMechanics 工具箱动力学仿真设置。
- 基于 SimMechanics 工具箱的车载 stewart 平台动力学建模与仿真。

13.1 SimMechanics 工具箱简介(中)

基于 SimMechanics 工具箱完成机械系统建模后,下一步进行的是进行仿真,仿真分为以下几个步骤。

1. 配置模型参数

这里的配置分为两部分。

1) 通过“Machine Environment”模块配置机械系统参数

每一个与地模块连接的刚体模型都需要配置该参数,且每一个“Machine Environment”模块只配置与其相连的刚体模型,能够配置的参数有动力学特性、机械维数、重力、精度、约束、运行分析模式和可视化定义等。

下面对这几个参数配置分别予以介绍。

(1) 定义重力。

机械系统中最为常见的参数是重力加速度,双击“Machine Environment”模块,打开参数配置对话框如图 13-1 所示。图中第一个参数即是重力场向量,默认是 $[0-9.81\ 0]\text{m/s}^2$,如果机械系统在地面上,则可采用默认值,而如果机械系统处于空间环境,如卫星、航天飞机等,则需要所处的轨道高度计算出重力场数值,并作相应的修改。

重力场向量除采用用户输入的常值外,还可以通过其他模块生成并输入,此时重力场向量可以是变化的值,这对某些特定场合的动力学分析是需要的,比如我国发射的“嫦娥”探月卫星从地面到月球的轨道转移过程中,地球重力场向量范围大,基于常值的动力学分析将会产生较大的偏差。

除重力场向量外,用户还可以对线安装偏差精度和角安装偏差精度等进行设置,如无特殊要求,用户可采用对话框给出的默认值。

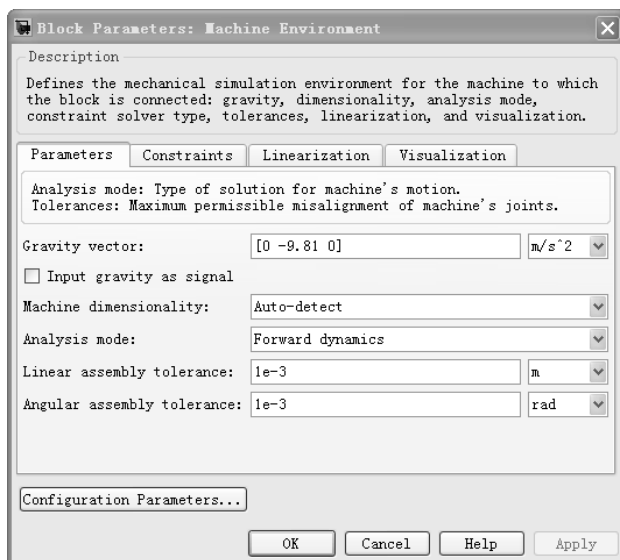


图 13-1 Machine Environment 模块参数设置界面

(2) 定义约束。

通过单击“Constraints”可打开约束定义窗口如图 13-2 所示。在该窗口中可以选择合适的约束解算器、设置是否处理奇异点和设置附加约束精度等。

其中约束解算器类型有三种：“Stabilizing”、“Tolerancing”和“Mechine precision”。如果选择“Tolerancing”，用户可以对相对精度和绝对精度进行设置，默认值都为 10^{-4} 。

设置是否处理奇异点是通过前面勾选框进行设置来实现的，如果选中，则在遇到奇异的时候，能够处理得更好。

附加约束精度设置也有两种模式：“Automatically select tolerance”和“Specify tolerance”。在“Specify tolerance”模式下，可以对相对精度进行设置，默认值是 10^{-14} 。

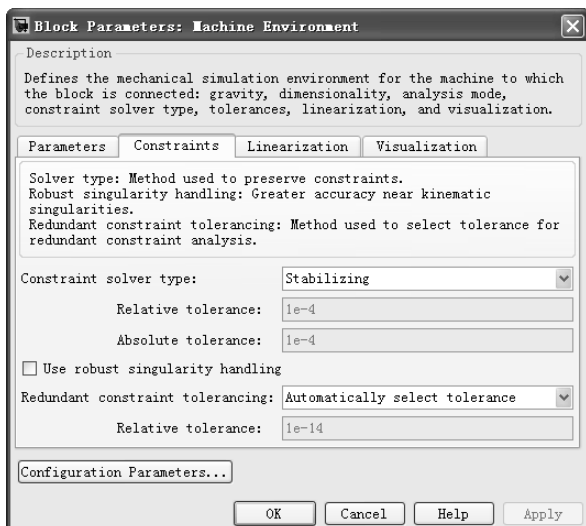


图 13-2 Machine Environment 模块限制设定

(3) 线性化设置。

单击“Linearization”可打开线性化设置对话框如图 13-3 所示，在该窗口中可以选择线性化类型以及线性化精度。其中线性化类型有“Fixed”和“Adaptive”两种，而线性化精度默认值为 10^{-5} ，用户可根据需要进行修改。

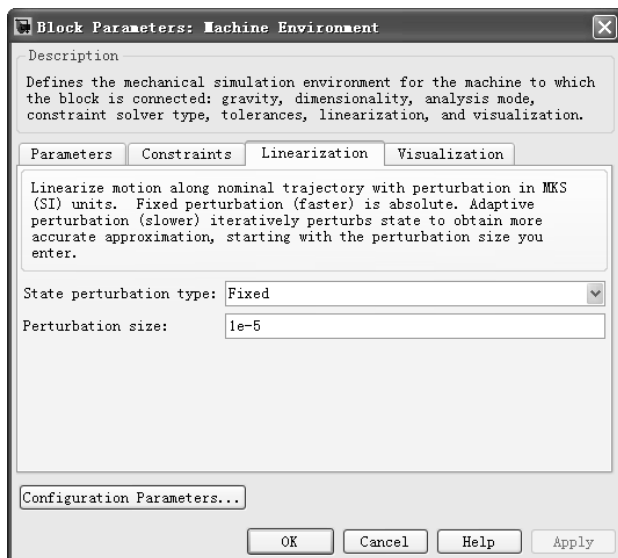


图 13-3 Machine Environment 模块线性化设置

(4) 可视化设置。

单击“Visualization”可打开可视化设置对话框如图 13-4 所示，该窗口用于设置可视化部分内容，关于机械系统可视化将在下一例详细介绍。

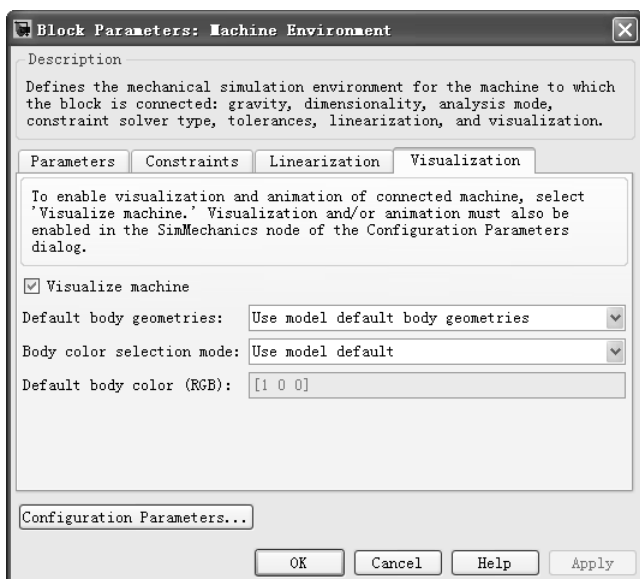


图 13-4 Machine Environment 模块可视化设置



2) 基于 Simulink 的仿真参数设置

通过单击菜单选项“Simulink|Configuration Parameters”或按快捷键“Ctrl+E”，可打开已搭建好的机械系统模型文件中的 Simulink 参数配置窗口，如图 13-5 所示。如图所示配置窗口中分为好几部分，这里主要介绍 Simscape 子标题 SimMechanics 中的参数设置。

图中右侧为对应 SimMechanics 仿真参数配置选项，分为两大部分：一部分是仿真诊断选项，包括发现多余约束是否警告、初始约束不稳定是否警告和切断连接是否标志；另一部分是可视化选项及参数设置，包括更新图标后是否显示机械对象、仿真中是否显示动画、是否只显示端口坐标系、默认的刚体颜色和默认的刚体外形。其中刚体颜色可通过三维 RGB 参数进行设置，默认的刚体外形可以选择基于体坐标系的凸包或基于质量特性的椭球。

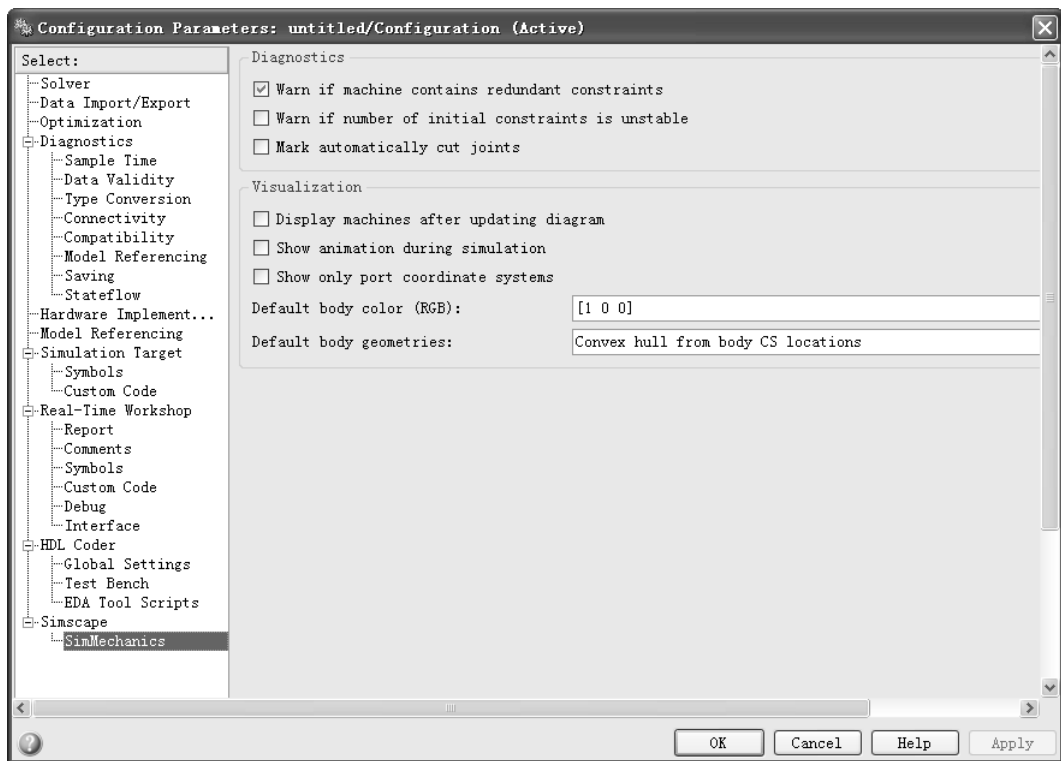


图 13-5 Simulink 仿真参数配置窗口

2. 动力学模型及分析过程

1) 一般机械对象的动力学

牛顿等式将力、加速度与机械对象的质量联系在一起，如式 (13-1) 所示：

$$F_A = m_A \frac{d^2 x_A}{dt^2} \quad (13-1)$$

欧拉等式将力矩、角加速度和机械对象的转动惯量联系在一起，如式 (13-2) 所示：



$$I_{ij} = \int_V dV \left[\delta_{ij} |r|^2 - r_i r_j \right] \rho(r) \quad (13-2)$$

两个方程的摄动形式如式 (13-3) 和式 (13-4) 所示：

$$F = m d^2 (\Delta \mathbf{x}) / dt^2 \quad (13-3)$$

$$\begin{cases} I_1 \Delta \dot{\omega}_1 - (\Delta \omega_2 \cdot \omega_3 + \omega_2 \cdot \Delta \omega_3) (I_2 - I_3) = \Delta \tau_1 \\ I_2 \Delta \dot{\omega}_2 - (\Delta \omega_3 \cdot \omega_1 + \omega_3 \cdot \Delta \omega_1) (I_3 - I_1) = \Delta \tau_2 \\ I_3 \Delta \dot{\omega}_3 - (\Delta \omega_1 \cdot \omega_2 + \omega_1 \cdot \Delta \omega_2) (I_1 - I_2) = \Delta \tau_3 \end{cases} \quad (13-4)$$

如果有约束方程，也需要得到此方程的摄动形式如式 (13-5) 所示：

$$g(x, \dot{x}, t) = 0, \frac{\partial g}{\partial x} \cdot \Delta x + \frac{\partial g}{\partial \dot{x}} \cdot \Delta \dot{x} = 0 \quad (13-5)$$

2) 基于动力学方程进行动力学分析过程

基于 MATLAB 的动力学方程允许两个方向的动力学分析。

(1) 正向动力学方式。在正向动力学方式中，通过给机械物体施加力或力矩，然后通过对产生的加速度或者角加速度进行两次积分，得到速度或者位置信息。

注意：在这种方式中，需要对初始状态进行设置。

(2) 反向动力学方式。在反向动力学方式中，首先给出随时间变化的位置和速度信息，然后通过微分得到力和力矩信息。

注意：基于反向方式得到力和力矩需要首先建立完整的动力学模型，该模型包含完整的位置、速度和加速度信息。

13.2 车载 stewart 平台建模与仿真

图 13-6 所示为一个 stewart 平台外形示意图，从图中可以看出平台分为上平面、上支柱、下支柱、下平面以及连接部件等。图 13-7 所示为平台结构示意图，图 13-8 为切断连接后的平台结构示意图，这是为了保证在 SimMechanics 中只有一个回路。

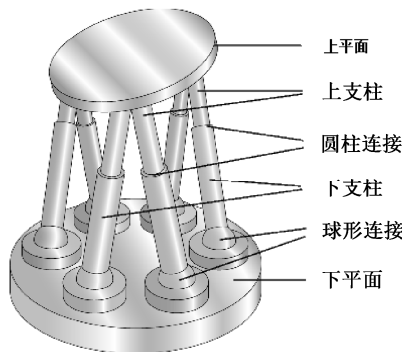


图 13-6 stewart 平台外形图



如图 13-9 所示为基于 SimMechanics 建立的 stewart 平台仿真模型。从仿真图中可以看出，模型分为四部分：参考信号生成部分、离散化部分、控制力生成部分以及平台动力学部分。其中前三部分都属于基于 Simulink 建立的模型，而平台动力学部分则基于 SimMechanics 建立。图 13-10 所示为平台动力学部分模块连接情况。

基于建立的仿真模型得到仿真结果如图 13-11 和图 13-12 所示。

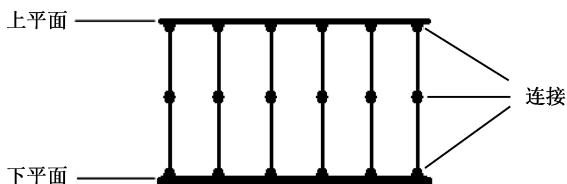


图 13-7 stewart 平台结构图

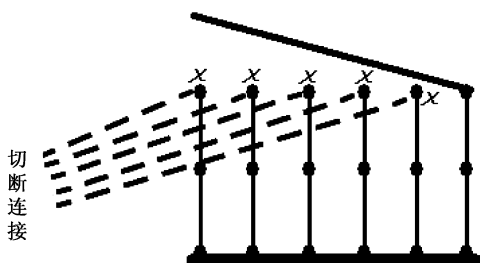


图 13-8 stewart 平台切断连接后结构图

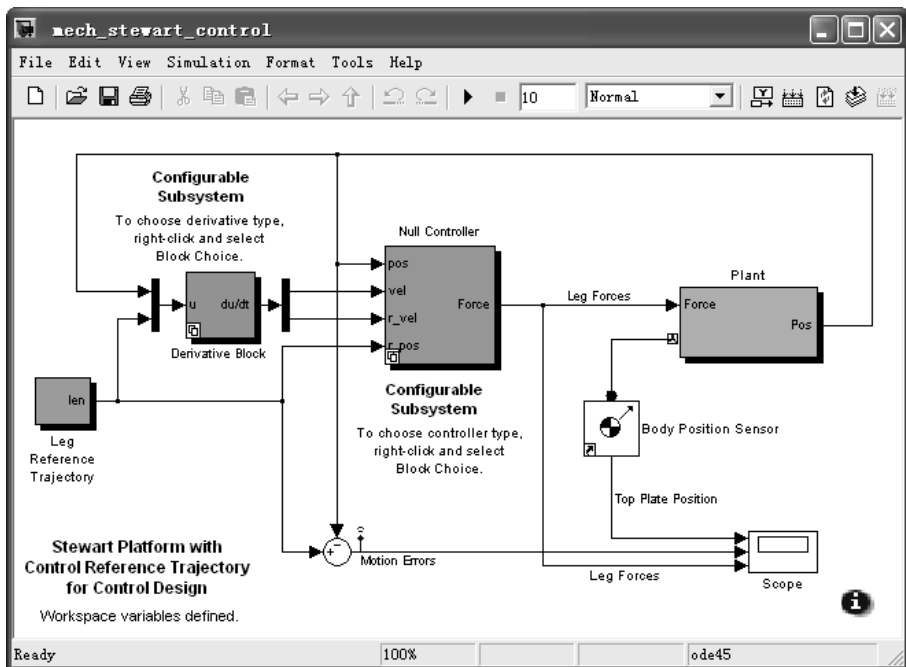
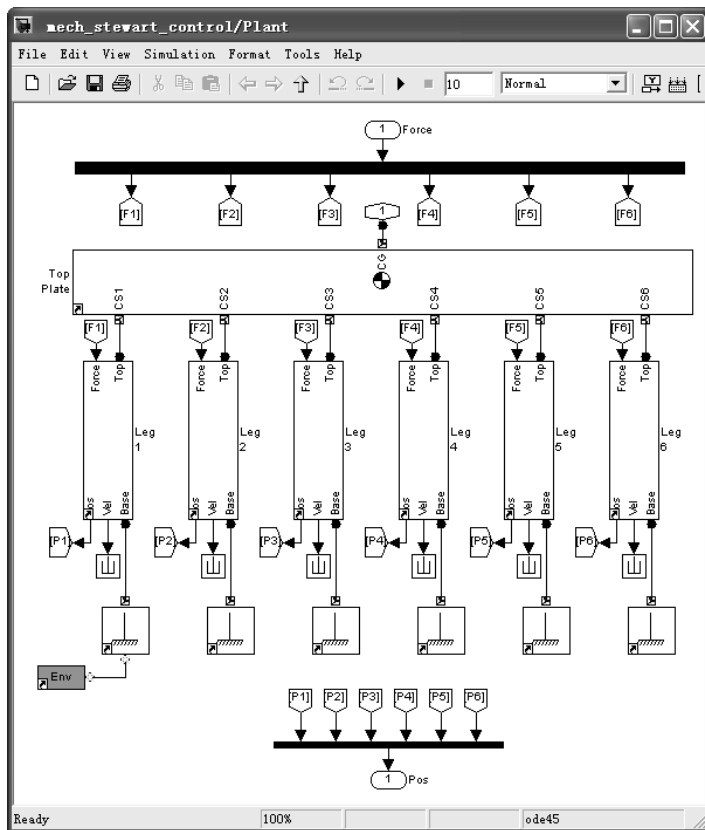
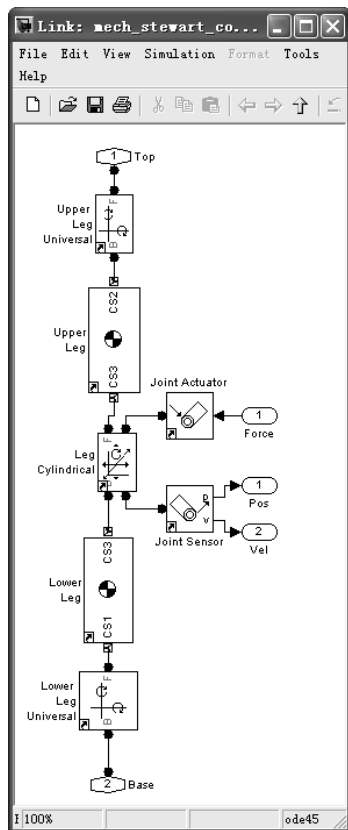


图 13-9 stewart 平台仿真图



(a)



(b)

图 13-10 stewart 平台动力学部分

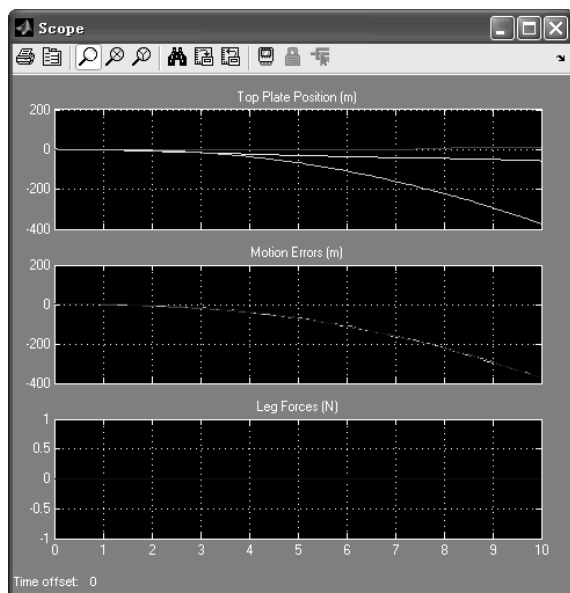


图 13-11 stewart 平台仿真结果 1

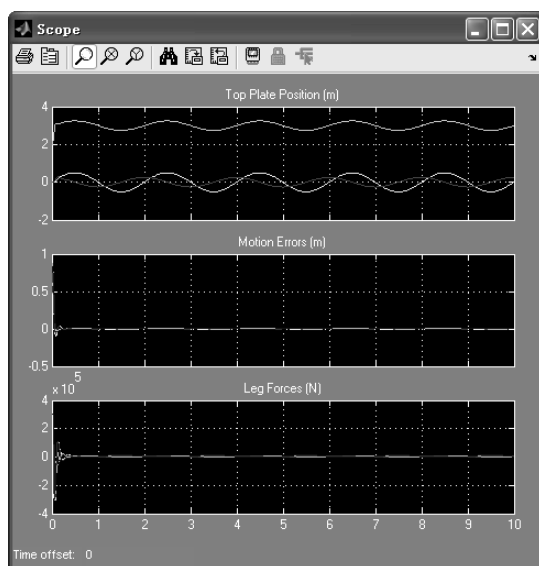


图 13-12 Stewart 平台仿真结果 2

13.3 本例小结

本例在引言基础上进一步介绍基于 SimMechanics 仿真的过程,通过建立 Stewart 平台实例,一方面演示了如何建立机械系统对象,另一方面演示如何实现基于 SimMechanics 建立模型以及基于 Simulink 模型之间信号的交换。



第 14 例 舰载雷达四杆机构建模与仿真

上一例介绍了基于 SimMechanics 工具箱进行机械对象的建模与仿真过程,由于可视化对于帮助理解机械运动具有重要的作用,因此本例进一步介绍 SimMechanics 工具箱的可视化部分。通过本例学习需要重点掌握 SimMechanics 工具箱三维显示系统设置过程。

14.1 SimMechanics 工具箱简介(下)

14.1.1 SimMechanics 工具箱可视化准备工作

进行 SimMechanics 三维仿真之前需要进行仿真设置,该设置通过单击 mdl 模块中菜单“Simulation”中的子菜单“Configuration Parameters”或者通过快捷键“Ctrl+E”打开仿真配置设置窗口,在窗口左侧列表中单击“Simscape”下的“SimMechanics”,其中关于可视化选项如图 14-1 所示。

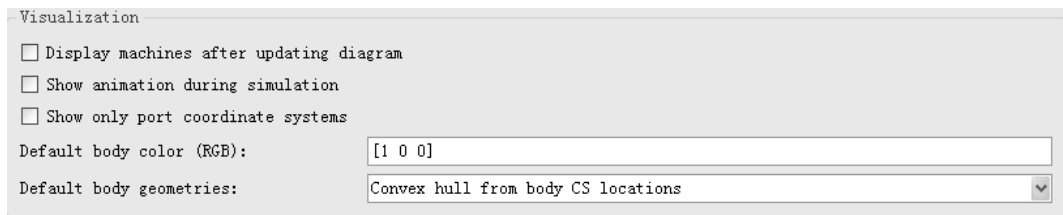


图 14-1 SimMechanics 工具箱可视化参数配置

从图中可以看出有三个选项。

- (1) 仿真之后进行三维显示,该选项用于仿真之后的静态三维观测。
 - (2) 仿真同时进行三维显示,该选项用于仿真同时的动态三维观测。
 - (3) 仅显示坐标系系统,该选项用于只需要观测坐标系变换的情况。
- 在仿真时根据需要勾选对应的选项。

除整体设置是否可视外,还可对每一个个体配置可视化属性,在建立好的机械模型文件中双击打开对应部件属性配置窗口,找到可视化配置部分如图 14-2 所示。

从图中可看出对每一个部件配置有两部分:一部分是物体外形,另一部分是物体颜色。在仿真时可根据需要选择合适的选项。

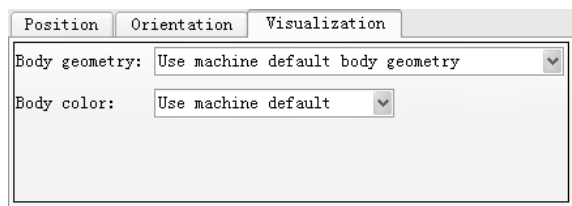


图 14-2 个体配置窗口

14.1.2 可视化仿真窗口介绍

可视化窗口是集成于机械工具箱中的一部分，通过它可以对机械进行可视化，并可以通过辅助工具对可视化进行设置并记录仿真过程。

可视化窗口的组成如图 14-3 所示。机械工具箱可视化窗口包含菜单栏、工具栏、显示窗口和状态栏四部分。其中菜单栏和工具栏用于操作可视化特征，显示窗口显示机械对象，状态栏显示当前状态。显示窗口还可细分为三维对象、背景和世界坐标系三部分。

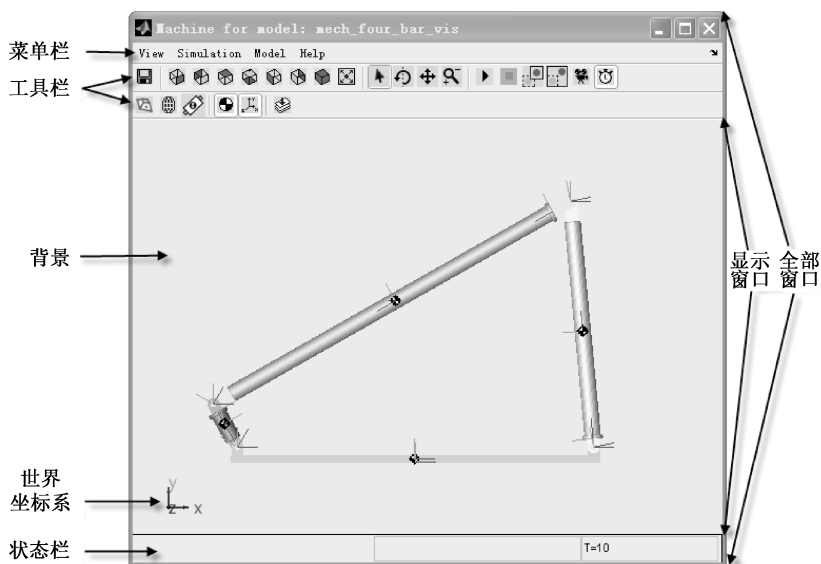


图 14-3 机械工具箱三维显示窗口组成

打开可视化窗口后，可以有两种方式来控制显示效果。

(1) 通过菜单栏。菜单栏分为可视、仿真、模型和帮助四部分，每一部分又包含子菜单，可根据需要选择设置。

(2) 通过工具栏。工具栏中的按钮单击后下沉表示选中状态，上升表示未选中。

菜单栏和工具栏绝大部分功能是相互重合的，通过工具栏能更方便快捷地完成可视窗口观测设置。

工具栏具体内容如图 14-4 所示，从图中可以看出工具栏分为以下几部分。

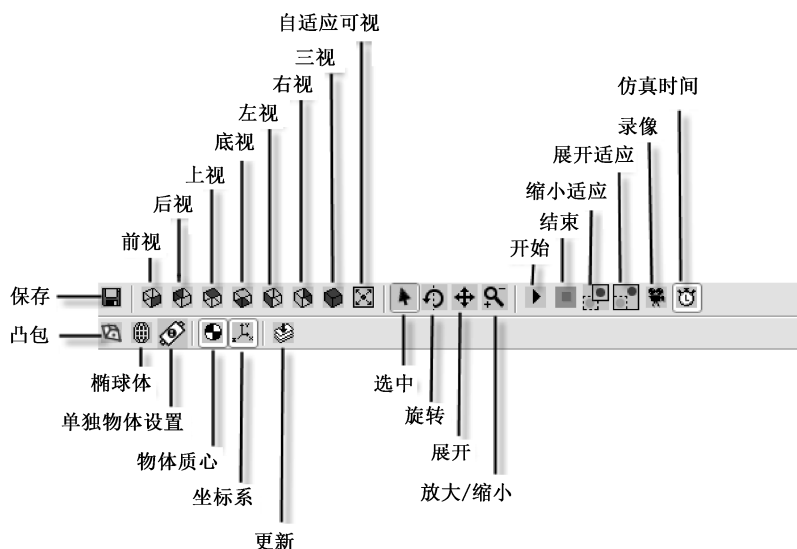


图 14-4 机械工具箱三维窗口工具栏组成

- (1) 保存当前设置按钮。
- (2) 视图设置，可设置正视、侧视和三视等观测角度按钮。
- (3) 放大、缩小、展开和旋转等设置按钮。
- (4) 仿真开始结束控制按钮。
- (5) 缩放适应和展开适应按钮。
- (6) 保存录像及显示仿真时刻按钮。
- (7) 设置物体形状的按钮，包括凸包、椭球体、重心、坐标系及更新等。

仿真开始后，为更好地对物体进行观测，需要进行合理的设置，下面介绍如何进行这些设置。

- (1) 设置背景颜色。

需要通过菜单栏来更改可视化窗口背景颜色，具体操作步骤如下：

单击“View|Change Background Color”打开调色板；

选中期望的颜色，单击“OK”按钮，则背景颜色修改成功。

- (2) 照相机投影、视场和视角。

照相机投影、视场和视角如图 4-15 所示。图中分别展示了可视化物体、平移、旋转和照相机平面大小。

- 正交投影。可视化窗口通过正交投影将三维物体投影到二维平面上。
- 视场是虚拟照相机能够看到的图像。虚拟照相机视角是指观测物体的点及方向，虚拟照相机视场的大小是有限的。

- (3) 自动设置照相机视场大小。

通过单击“自适应可视”按钮可使物体在视场中全部出现且与窗口大小匹配。图 14-6 和图 14-7 展示了自适应可视前后视场大小的对比情况。

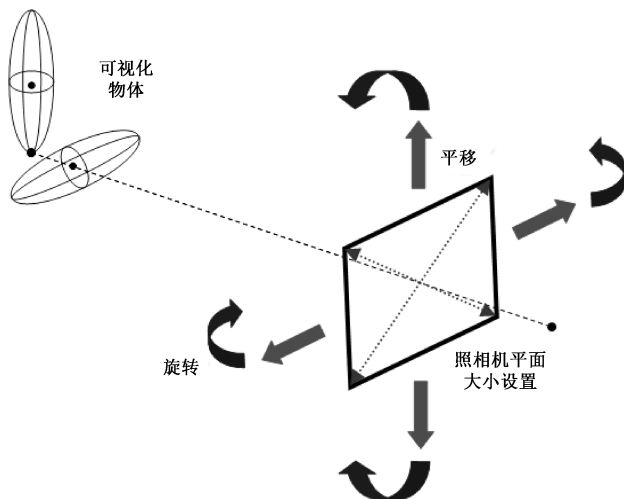


图 14-5 照相机投影、视场和视角

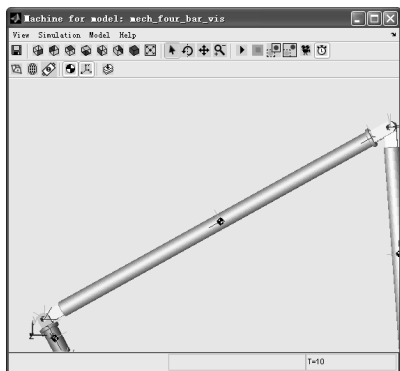


图 14-6 自适应大小之前

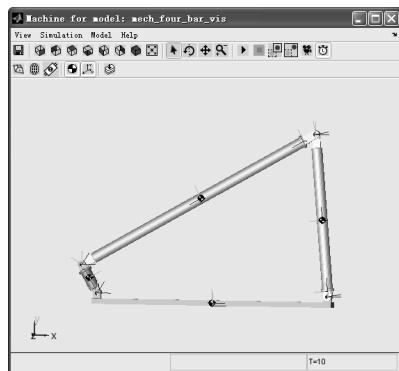


图 14-7 自适应大小之后

(4) 控制照相机视角。

通过在显示窗口中，左键单击并拖拽完成控制视角动作，控制的类型有以下三个：

- 放大缩小。通过改变照相机帧大小来改变视场中物体的大小，视场中没有物体是不可以改变帧大小的。
- 平移是不改变照相机方向，而只是在一个固定平面中做垂直方向、水平方向或两者结合的移动，照相机的视场也随着改变。
- 旋转是将照相机固定于一点，改变照相机观测角度，从而使照相机观测的视场发生变化。

14.2 舰载雷达四杆机构仿真

平面四杆机构是由四个刚性构件用低副链接组成的，各个运动构件均在同一平面内运动的机构。所有运动副均为转动副的四杆机构称为铰链四杆机构，它是平面四杆机构

的基本形式，其他四杆机构都可以看成是在它的基础上演化而来的。选定其中一个构件作为机架之后，直接与机架链接的构件称为连架杆，不直接与机架链接的构件称为连杆，能够做整周回转的构件被称作曲柄，只能在某一角度范围内往复摆动的构件称为摇杆。如果以转动副连接的两个构件可以做整周相对转动，则称之为整转副，反之称之为摆转副。铰链四杆机构中，按照连架杆是否可以做整周转动，可以将其分为三种基本形式，即曲柄摇杆机构，双曲柄机构和双摇杆机构。

图 14-8 所示为四杆机构外形图，图 14-9 为其结构图，图 14-10 和图 14-11 所示为环境模型及地模型参数设置，图 14-12 为仿真结果，图 14-13 所示为四杆机构三维模型。

单击仿真，可以看出图 14-13 中四杆机构随着仿真时间在运动，读者也可以在三维显示窗口中修改设置，体验 13.1 节中介绍的功能。

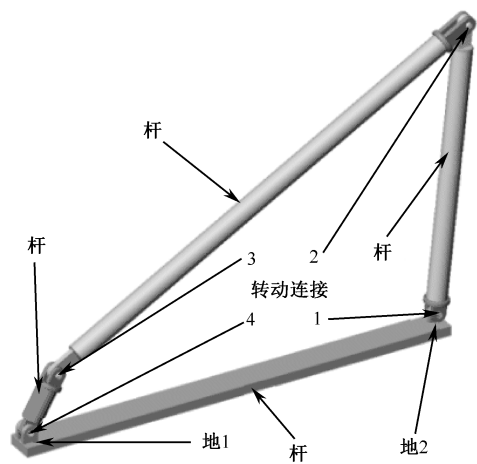


图 14-8 四杆机构外形图

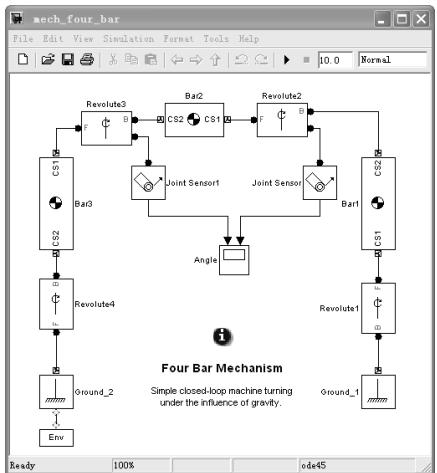


图 14-9 四杆机构结构图

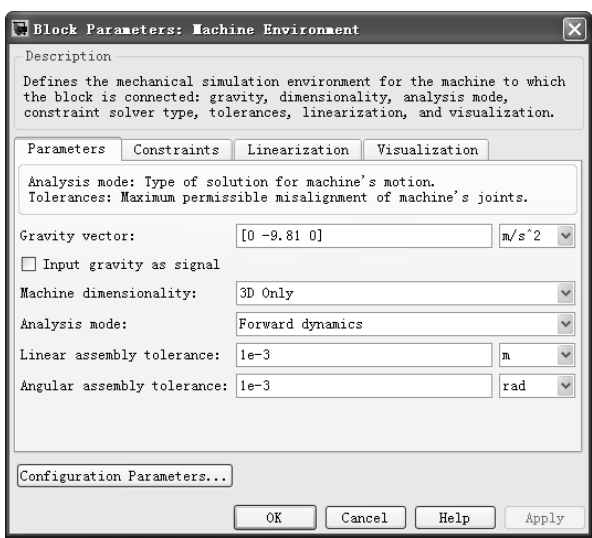


图 14-10 机械环境设置

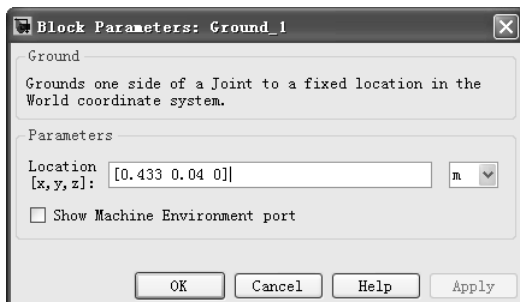


图 14-11 地模型设置



图 14-12 仿真结果

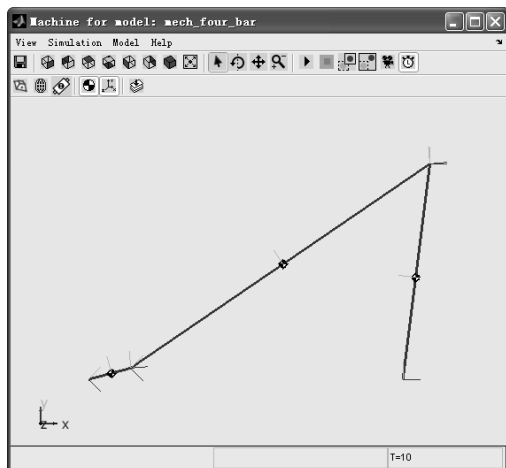


图 14-13 四杆机构三维图

14.3 本例小结

本例以四杆机构为例介绍了 SimMechanics 工具箱三维显示系统，基于该系统，能够更加直观地观测机械对象运动，用户可根据需要选择是否显示三维动画以及选择在仿真运动时显示还是在仿真结束后显示。



第 15 例 基于 SolidWorks 的机载 Stewart 平台建模与仿真



在第 13 例和第 14 例中分别介绍了基于 MATLAB 机械工具箱建立机械对象动力学模型及基于自带的可视化窗口建立三维可视化模型过程，这里介绍另外一种建立机械对象动力学模型和三维可视模型的方法。该方法首先在通常使用的 CAD 辅助软件（SolidWorks、Pro/E 等）里面建立机械系统模型，然后通过接口转换成 MATLAB 机械工具箱能够读取的文件并在其中仿真。这种方法既能够采用传统 CAD 便利的建模工具完成机械对象建模工作，又能将其迅速地转换成 MATLAB 能够辨识的对象，并利用 MATLAB 丰富的数学工具完成数学仿真。

通过本例学习，应该掌握以下两点：

- 基于一般 CAD 工具建立模型与 MATLAB 中 SimMechanics 模型转换过程。
- 基于 SolidWorks 模型与 SimMechanics 模型转换过程。

15.1 从 CAD 建模工具中输入模型

15.1.1 转换步骤

SimMechanics 可以通过 XML 文件自动生成能够运行的仿真模型。这使得可以通过在外部建立 XML 文件对机械机构，自由度和几何等信息进行定义，然后传送至 SimMechanics 进一步使用。

建立 XML 的一个方法是通过计算机辅助装配（CAD）工具实现。CAD 工具允许建立机械系统零件或装配的几何模型。但是基于 CAD 工具建立的模型存在一个问题，即它只包含几何和连接信息，不包含控制模块，这可以通过在 Simulink 和 SimMechanics 中完成。通过结合 CAD 工具和 SimMechanics 可更加迅速和更好地完成对机械系统的仿真。

SimMechanics 提供的接口是 CAD 工具转换到 SimMechanics 模型的关键。接口在 CAD 工具生成 XML 文件的同时，还生成 STL 文件。该 STL 文件包含机械的几何和形状信息，基于该文件可进行机械系统的三维仿真。

由 CAD 工具转换至 SimMechanics 的过程如图 15-1 所示。

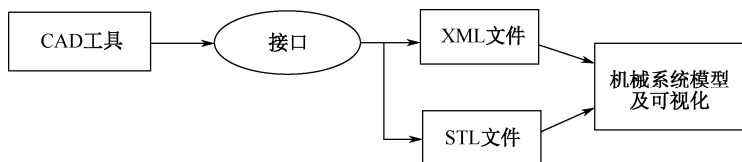


图 15-1 CAD 工具转换至 SimMechanics 的过程

由图 15-1 可以看出转换分为两个步骤：第一步是从 CAD 工具中得到 XML 文件和 STL 文件；第二步是从 XML 文件和 STL 文件生成机械系统模型。

其中第一步从 CAD 工具生成 XML 文件和 STL 文件需要两个条件：一是安装 CAD 工具；二是需要安装 MATLAB 机械工具箱接口文件，该接口文件可从 MATLAB 官网下载，下面将介绍使用方法。

1. 生成 XML 和 STL 文件

从 CAD 工具生成 XML 和 STL 文件首先需要安装接口文件，该接口文件名称为“SimMechanics Link”，安装此文件的过程如下。

(1) 打开 http://www.mathworks.com/products/simmechanics/download_smlink.html，可下载 SimMechanics Link，该处提供了多个版本，用户可以根据自己系统及 MATLAB 版本选择合适的 SimMechanics Link。

(2) 根据网站提供的信息下载相应的文件，文件包含两个：一个是压缩文件，另一个是 m 文件。例如“smlink31.win32.zip”和“install_addon.m”，用户根据版本不同，第一个文件名存在不同。

(3) 打开 MATLAB，并输入“install_addon('smlink31.win32.zip')”，完成安装。

2. 基于 XML 生成机械系统模型

CAD 工具中包含足够的零件和约束的信息，可以基于 CAD 生成一个机械系统模型。基于 CAD 生成一个 XML 文件后，SimMechanics 可基于该文件建立框图模型。

通过命令“mech_import”完成由 XML 文件生成机械模型，在 MATLAB 命令行中输入“mech_import”可得到输入 XML 文件的对话框，如图 15-2 所示。

由图可以看出对话框分为两栏，第一栏是输入文件对应的多个设置，分为四个部分：

- XML 文件输入地址。
- 选择输入文件的类型，也是分为四类，分别是：输入到新模型、更新已有模型、增加到已有模型，替换已有模型。

- 输出模型的地址（适用于有输出模型的模式）。

- 指定被替换的分系统，适用于替换已有模型这个模式。

对话框中第二栏是一些高级选项，如图 15-3 所示，也是分为四个部分：

- 指定模型。其中有三个子选项，分别是：无指定模型，将输入模型连接于顶层或固连于焊接于一起的物体上以及焊接于一起的物体属于同一组。

- 名称分布。主要设置机械物体对应名称显示。

- 更新选项。用于对更新模型进行更多的设置。

- 备用模型。设置备用模型的模式和位置。

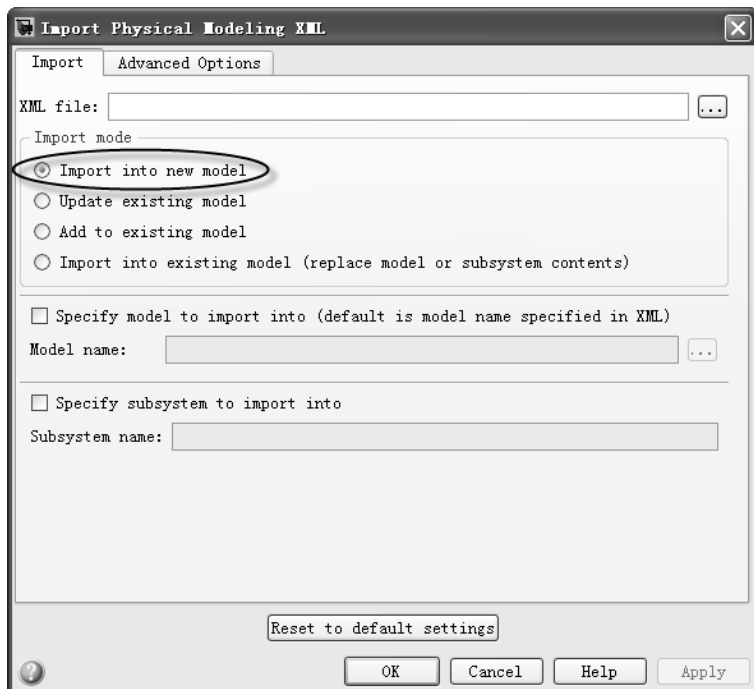


图 15-2 输入 XML 文件对话框

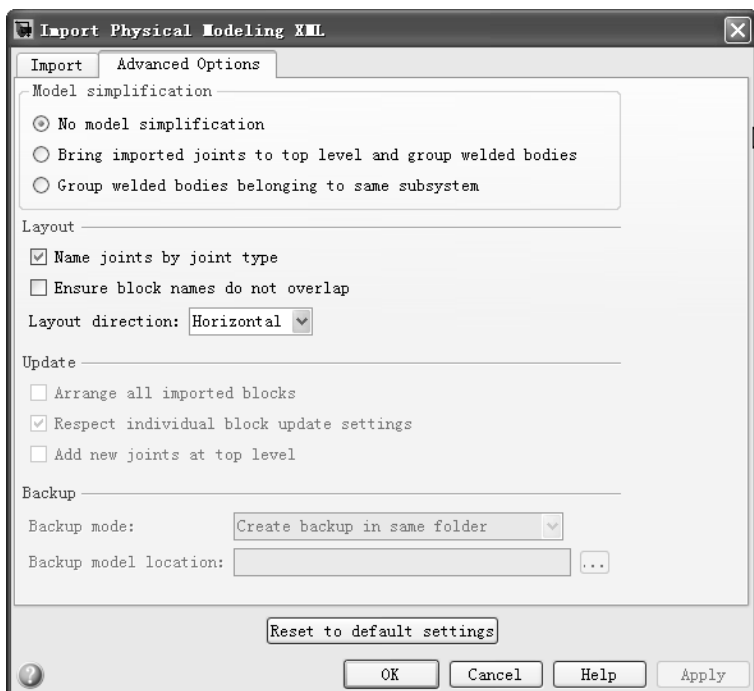


图 15-3 输入 XML 对话框高级设置



15.1.2 生成模型特性

下面介绍生成模型的一些特性，这些特性包括共同的特性以及不同的特性。

首先研究共同的特性，共同特性共有以下八种。

注意：这里的讨论是基于采用默认设置转换生成的模型。

转换生成模型的绝大部分特性与 Simulink 模型的默认属性是一致的。下面介绍一些基本的共同属性：

(1) 机械环境模型、地模型、体模型和连接件模型，这些模型中相同的属性有：

- 转换生成的模型中都是只有一个环境模型和一个地模型。
- 其他自动生成的模型是体模型和连接件模型。

(2) 体坐标系模型和地坐标系模型都是至少需要连接两个坐标系系统，连接件的一头连接一个。

(3) 未连接的坐标系系统没有约束。

(4) 地模型首先转换过来，且没有质量及惯性等。

(5) 包含子系统。

(6) 除连接属性外，还包含三维信息。

(7) 同时支持部件和整体机械模型可视。

注意：为保证可视化，STL 文件在 MATLAB 工作路径或者设置 STL 文件的完整路径。

(8) 生成的地模型达标世界坐标系的原点，这个原点对应原 CAD 模型的原点。

下面给出一个转换实例。

转换实例如图 15-4 所示，由图可以看出，转换之后的模型与直接基于机械工具箱生成的模型具有很多共同的特点。

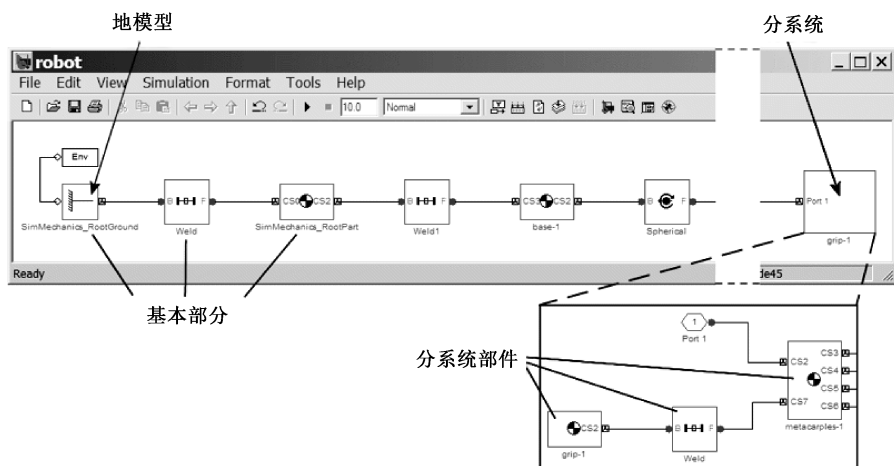


图 15-4 转换实例

除共同的属性外，转换生成的模型与直接生成的模型也存在不同，这些不同有以下几项。

(1) 非默认的外观。

表 15-1 列出了在外观方面可能非默认的选项。

表 15-1 非默认外观选项

默认外观属性	非默认外观属性
短连接件模型名称	长连接件模型名称
模型名称重叠	模型名称不会重叠
模型列表首先从左至右，然后从上至下	模型列表首先从上至下，然后从左至右

(2) 非默认的继承性。

表 15-2 列出了在继承性方面可能非默认的选项。

表 15-2 非默认继承性选项

继承性分类	默认选项	非默认选项 A	非默认选项 B
非固定连接件位置	在分系统层	在最高层	与 A 相同且默认
固定连接固体组合	非组合且独立分布	组合成整体	与 A 相同且非默认

(3) 坐标系系统。

转换模型与自生成模型在坐标系系统不同的是：

- 转换生成坐标系与原 CAD 模型中的坐标系不是一一对应的。
- 转换之后的模型还可以往里面添加坐标系系统。

15.1.3 转换后模型修改

转后之后的模型并不能完全满足仿真的需求，需要对其进行适当的修改。修改包括以下几个方面：删除不需要的模块、增加适当的坐标系系统、限制自由度和添加驱动刚体的力或力矩等。下面对这几方面分别予以介绍。

提示：在对转换模型进行修改之前，可保存一个副本。

1. 删除不需要的模块

转换生成的某些模块对于仿真来说并不都是需要的，为简化仿真，可以删去这些模块。尤其对于固连模块，在以下几种情况下可以删掉：

- 如果固连体只通过一个连接件连接其他模块，此时可以删掉该模块，而将连接件模块直接连接于地模块。
- 如果固连体通过多个连接件连接其他模块，此时不可以删掉该模块，但可以通过多个地模型进行替换。



2. 增加部分坐标系系统

除转换生成的坐标系系统外，还可以添加自定义的坐标系系统。如果需要修改基于 CAD 模型生成系统，来增加约束、驱动、执行机构和敏感器等，此时就需要增加额外的坐标系系统来连接相应的模块。

3. 增加约束

约束用于限制刚体的自由度，有时需要针对转换生成的系统添加额外的约束来使机械系统与实际对象特性相符合。

4. 增加执行机构

转换生成的模型中并不包含执行机构，为仿真机械系统动力学特性，需要添加特定的执行机构，使系统在力或力矩驱动下运动，并观察仿真结果进行分析。

5. 更改生成物体的外观和颜色

转换生成的模型自动设定物体的外观和颜色，而这些设定不一定满足用户的要求。为制定符合要求的三维模型，用户可以对外观和颜色进行修改。

15.2 基于 SolidWorks 的 stewart 平台三维模型转换

图 15-5 所示为基于 SolidWorks 的 stewart 平台模型，由图可知其分为五部分：上平面、上支柱、下支柱、下平面和部分连接件。按照 15.1 节给出的转换步骤，可以得到如图 15-6 所示的结构模型和图 15-7 所示的三维模型。

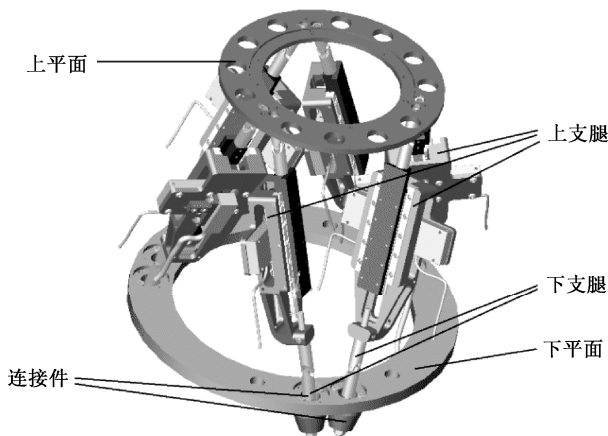


图 15-5 基于 SolidWorks 的 stewart 平台模型

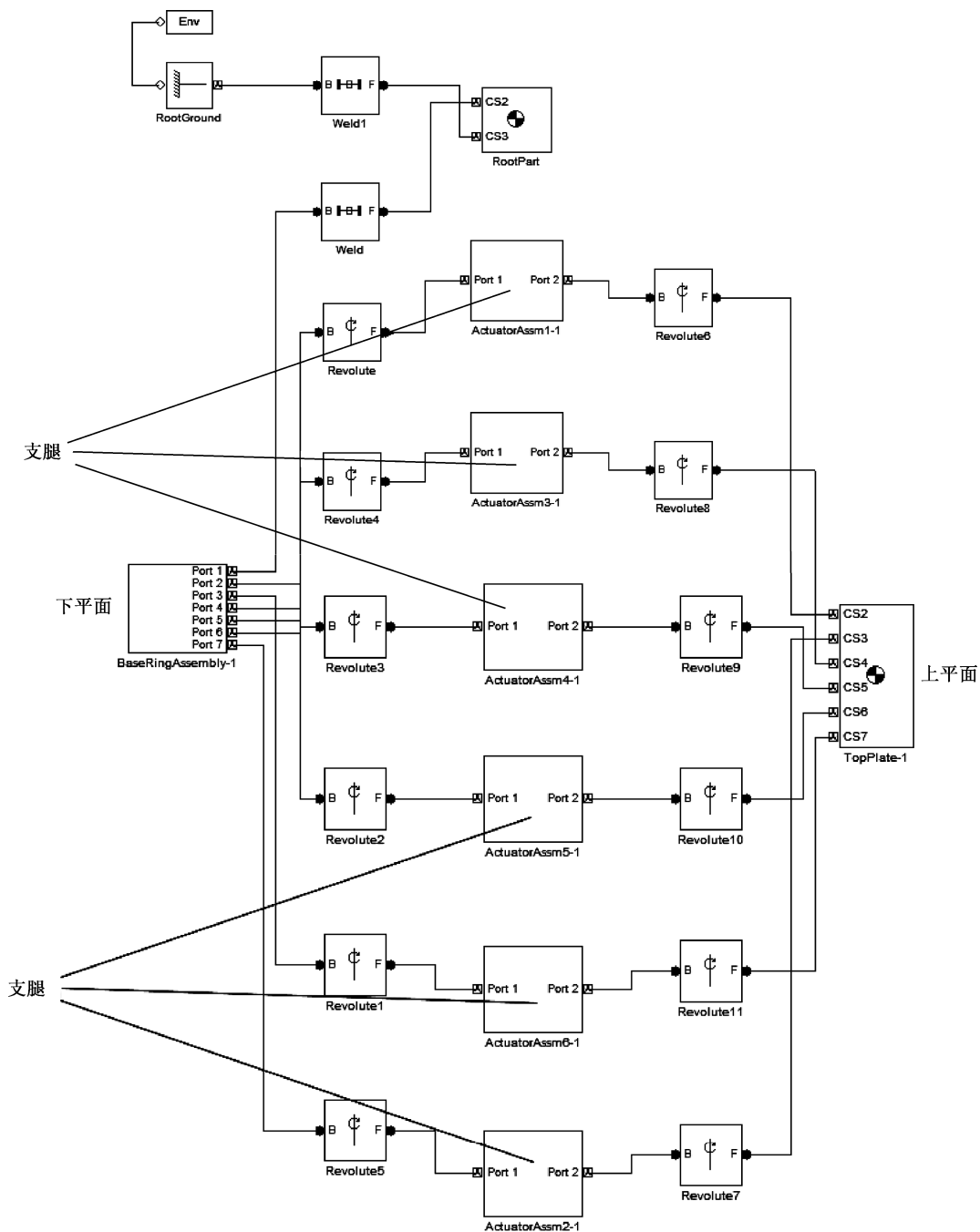


图 15-6 stewart 平台转换之后的结构模型

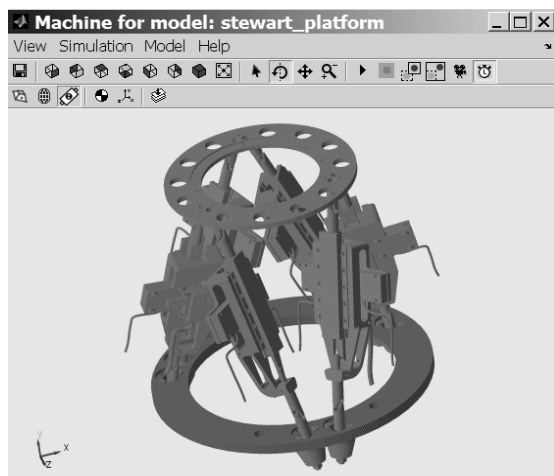


图 15-7 stewart 平台转换之后的三维模型

15.3 本例小结

本例介绍了由一般 CAD 工具生成的机械模型转换为 SimMechanics 模型的过程，该转换将有助于重用以往基于一般 CAD 工具建立的机械模型，从而能够极大地提高效率，并将 CAD 工具三维建模能力与 MATLAB 计算能力结合起来。



第 16 例 卫星三维建模与有限元分析



本例以卫星为对象,基于 MATLAB 建立能够进行卫星三维建模与有限元分析的 GUI 界面,通过本例学习,应掌握以下两点:

- 基于 M 语言建立 GUI 界面过程。
- 有限元分析过程。

16.1 基于 M 语言的 GUI 界面设计

图形用户界面 (GUI) 是通过一个或多个,包含控件调用的组件,使用户能够执行交互式任务的显示窗口。GUI 用户不必创建脚本或在命令行键入命令来完成任务,也不需要了解如何执行任务的细节。用户可以借助 GUI 界面迅速方便地完成任务。

GUI 界面中包括菜单,工具栏,按钮,单选按钮,列表框,滑块等。基于 MATLAB 的 GUI 可以执行任何类型的计算,例如读取和写入数据文件,沟通与其他图形用户界面,显示数据表等。

基于 MATLAB 创建 GUI 界面有两种方式:一种是基于 M 语言;第二种是基于 MATLAB 提供的 GUIDE 工具。这里介绍基于 M 语言创建 GUI 界面的方式,在第 20 例中将介绍基于 GUIDE 创建 GUI 界面的方式。

基于 M 语言创建 GUI 界面主要分以下几个步骤。

1. 设计 GUI 界面

创建 GUI 之前,首先要明确希望 GUI 能够达到什么功能,常用的方法是作出草图,完成 GUI 初步设计。

注意: MATLAB 已经提供了一些标准的 GUI,如错误提示对话框 (errordlg)、帮助对话框 (helpdlg)、消息对话框 (msgbox) 和输入对话框 (inputdlg) 等。用户在设计时适当参考或借用这些标准 GUI 有助于节约时间,提高效率。

2. 建立 GUI 文件

由于是基于 M 语言创建 GUI,所以首先要建立一个文件。该文件为函数文件,形式与一般的函数文件类似。下面给出示例函数文件 mygui:



```
function varargout = mygui(varargin)
end
```

3. 建立图形窗口

创建 GUI 函数文件之后，下面就需要补充具体的 GUI 内容。GUI 是图形交换界面，所以首先建立图形窗口。创建图形窗口命令如下：

```
fh = figure('Visible','on','Name','My GUI',...
            'Position',[360,100,450,285]);
```

该命令中 f 为图形窗口句柄。句柄是一个标识符，用于标识对象，对象可以是函数、图形窗口等。它就像我们的姓名一样，用来区别和标识某一特定对象。图形窗口句柄就是用来标识特定的图形窗口。

注意：在基于 M 语言建立 GUI 中，图形窗口句柄是一个非常重要的概念，下面将进一步阐述。

创建图形窗口之后，可同时设置图形窗口的属性，表 16-1 列出了图形窗口中常见的属性。

表 16-1 图形窗口属性

属 性	值	描 述
菜单 (MenuBar)	figure/none, 默认是 figure	显示或隐藏菜单栏, 如果是 none, 则隐藏, 反之显示
名称 (Name)	String	图形名称
名称目 (NumberTitle)	on/off	如果为 off, 则不显示图形名称, 反之显示
位置 (Position)	[至左边距离, 至底边距离, 宽, 高]	描述图形在显示器中的位置
改变大小 (Resize)	on/off, 默认是 on	决定用户是否可以用鼠标改变图形大小
工具栏 (Toolbar)	auto/none/figure, 默认是 auto	如果是 none, 则不显示, 如果是 figure 则显示, 如果是 auto 则由 uicontrol 决定
单位	pixels, inches, centimeters, characters, normalized, points, 默认是 pixels.	图形显示单位
可视 (Visible)	on/off, 默认是 on	决定图形是否显示

4. 加入图形窗口组件

上一步中建立了图形窗口，下面在图形窗口中添加窗口组件。窗口组件是 GUI 中完成特定功能的模块。MATLAB 中窗口组件分为四类：

- 用户交互控制组件，如按钮和滑块。
- 容器组件，如面板和按钮组等。



- 图形组件，用于包含作图曲线和图像等。
- ActiveX 组件，用于和 Windows 平台交互。

说明：MATLAB 同样提供了一些标准组件模型，可根据需要选用。通过在 MATLAB 帮助文档中输入“Predefined Dialog Boxes”可查看。

下面给出一些常用的标准组件命令，如表 16-2 所示。

表 16-2 MATLAB 常用标准组件命令

组 件 对 象	函 数 命 令	描 述
图形组件 (Axes)	Axes	用于显示曲线或图像
按钮框组件 (Button Group)	Uibuttongroup	用于包含多个按钮
复选框组件 (check Box)	uicontrol	通过回调函数可实现选中与未选中代表两种状态
文本框组件 (Edit Text)	uicontrol	用于输入字符或文字，GUI 将其作为字符串处理
列表框组件 (List Box)	uicontrol	列出多个选项，根据选择不同返回不同状态
面板组件 (Panel)	uipanel	面板用于存放组件，通常通过面板分类使 GUI 整齐美观
按钮 (Push Button)	uicontrol	当按钮按下时，将产生一个动作，通过回调函数将产生相应事件

5. 创建菜单

创建菜单通过命令“uimenu”完成，其完整调用格式如下所示：

```
mh = uimenu(parent,'PropertyName',PropertyValue,...)
set(fh,'MenuBar','figure');    % 显示 MATLAB 标准菜单
set(fh,'MenuBar','none');      % 隐藏菜单
```

“uimenu”命令中部分属性如表 16-3 所示。

表 16-3 菜单属性

属 性	值	描 述
附加菜单 (Accelerator)	字母	增加附加菜单
选中 (Checked)	off/on，默认是 off	菜单选中显示
使能 (Enable)	off/on，默认是 on	菜单能否被选中
句柄是否可见 (HandleVisibility)	off/on，默认是 on	描述一个句柄在父对象中是否可见
标签 (Label)	字符	菜单名称
位置 (Position)	系数，默认是 1	在菜单中的位置
分隔 (Separator)	off/on，默认是 off	对菜单进行分隔

下面给出基于菜单属性建立菜单例子：



```
mh = uimenu(fh, 'Label', 'My menu');  
eh1 = uimenu(mh, 'Label', 'Item 1');  
eh2 = uimenu(mh, 'Label', 'Item 2', 'Checked', 'on');
```

命令语句中 fh 是图形窗口句柄,通过该句柄使菜单在该图形窗口中显示。mh 是菜单句柄,通过该句柄建立两个菜单项。

注意:上述建立的菜单,如果图形窗口采用的是标准菜单,则在后面添加一系列菜单;如果是非标准菜单,则新建一系列菜单。

6. 建立工具栏

建立工具栏用命令“uitoolbar”、“uipushtool”和“uitoggletool”完成,其完整的调用格式是:

```
tbh = uitoolbar(h, 'PropertyName', PropertyValue, ...)  
pth = uipushtool(h, 'PropertyName', PropertyValue, ...)  
tth = uitoggletool(h, 'PropertyName', PropertyValue, ...)
```

工具栏命令中常用的属性如表 16-4 所示。

表 16-4 工具栏常见属性

属 性	值	描 述
工具栏图标 (CData)	从 0 至 1 的三维数组	RGB 数组,用于建立工具栏图标
句柄可见性 (HandleVisibility)	off/on, 默认是 on	决定工具栏句柄在父对象中是否可见
分隔符 (Separator)	off/on, 默认是 off	分隔工具栏
状态 (State)	off/on, 默认是 off	工具栏工具是否下沉
动作字符 (TooltipString)	字符	工具栏工具产生动作时是否显示字符

基于工具栏建立命令建立工具栏例子如下:

```
% 建立工具栏  
th = uitoolbar(fh);  
% 增加一个点击式工具  
a = [.20:.05:0.95]  
img1(:, :, 1) = repmat(a, 16, 1)'  
img1(:, :, 2) = repmat(a, 16, 1);  
img1(:, :, 3) = repmat(flipdim(a, 2), 16, 1);  
pth = uipushtool(th, 'CData', img1, ...  
    'TooltipString', 'My push tool', ...  
    'HandleVisibility', 'off')  
% 增加一个切换式工具  
img2 = rand(16, 16, 3);  
tth = uitoggletool(th, 'CData', img2, 'Separator', 'on', ...  
    'TooltipString', 'Your toggle tool', ...  
    'HandleVisibility', 'off')
```



7. 编写回调函数

所谓的回调函数，就是空间接受到某个 Windows 消息，比如说鼠标的单击，双击和移动，以及键盘输入的时候，对所希望进行的操作指定的函数，该函数不会主动运行，是由主控程序调用的。主控程序一直处于前台操作，它对各种消息进行分析，排队和处理，最后去调用指定的回调函数，执行完毕之后控制权又回到主控程序，由主控制程序在必要的时候调用执行，这就是回调函数的本意。

常用的回调函数有以下几个：

- Callback：与空间相关的标准回调函数。不同的控件可以响应不同的事件，尽管有相同的属性名，但是其实现的功能却因控件的不同而不同。如按钮的 Callback 是由于鼠标的一次单击引起的，而 PopupMenu 则是鼠标单击下拉按钮，然后在列表中单击一个条目之后发生的。
- ButtonDownFcn：鼠标左键在控件上按下。
- CreateFcn：控件生成。
- DeleteFcn：控件删除。

Figure 控件有自己的许多特殊回调函数：

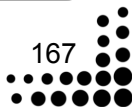
- CloseRequestFcn：Figure 关闭时执行。
- KeyPressFcn：有键按下。
- ResizeFcn：用户调整 Figure 的大小。
- WindowButtonDownFcn：在 Figure 的空白处单击。
- WindowButtonMotionFcn：鼠标在 Figure 上方移动时。
- WindowButtonUpFcn：在 Figure 上单击鼠标又抬起之后。属性 Callback 中的字符串在默认情况下为字符串<automatic>，而其他回调函数为空白。

16.2 卫星三维建模与有限元分析

有限元法 (Finite Element Analysis, FEA) 的基本概念是用较简单的问题代替复杂问题后再求解。它将求解域看成是由许多称为有限元的小的互连子域组成，对每一单元假定一个合适的 (较简单的) 近似解，然后推导求解这个域总的满足条件 (如结构的平衡条件)，从而得到问题的解。由于大多数实际问题难以得到准确解，而有限元不仅计算精度高，而且能适应各种复杂形状，因而成为行之有效的工程分析手段。利用有限元法对卫星结构和热分布等进行分析是卫星工程中行之有效的手段。

为建立卫星三维建模与有限元分析工具，同时演示基于 M 语言建立 GUI 过程，这里给出基于 MATLAB 建立的卫星三维建模与有限元分析软件代码 (由于篇幅所限，这里只给出部分代码，完整代码请见提供下载的源代码)：

```
function pdetool(action,flag) % 函数文件
if nargin<1
    action='initialize'; % 如果无输入，则为初始化
```





```
end
global pde_fig      % 给出四个全局变量，以便于不同句柄对象间数据传递
global ax
global shuju
global num

pde_fig=findobj(allchild(0),'flat','Tag','PDETool'); % 寻找对象并对句柄
赋值

if strcmp(action,'initialize')
    num = [1 0];
    if(~isempty(pde_fig))      %如果已存在图形窗口，则以此建立 GUI
        set(0,'CurrentFigure',pde_fig);
        Figpos = get(pde_fig,'Position');
        set(pde_fig,...
            'Colormap',gray(20),...
            'HandleVisibility','on')
        delete(allchild(pde_fig))
        refresh(pde_fig)
    else                      % 如果不存在图形窗口，则新建窗口并建立 GUI
        ScreenUnits = get(0,'Units');
        set(0,'Unit','pixels');
        ScreenPos = get(0,'ScreenSize');
        set(0,'Unit',ScreenUnits);
        Figpos=[.15 .15 .8 .75].*ScreenPos+[0.1*ScreenPos(3:4) 0 0];
        pde_fig=figure(...
            'Color','w',...
            'Colormap',gray(20),...
            'Position',Figpos,...
            'Interruptible','on','CloseRequestFcn','pdetool(''exit'')');
    end

    set(pde_fig,'Pointer','watch')
    drawnow

    flags=zeros(7,1);

    c='1.0'; a='0.0'; f='10.0'; d='1.0'; tlist='0:10'; u0='0.0';...
    ut0='0.0'; r='[0 100]'; rtol='0.01'; atol='0.001';
    params=str2mat(c,a,f,d);
    setappdata(pde_fig,'currparam',str2mat(c,a,f,d))

    setappdata(pde_fig,'timeeigparam',str2mat(tlist,u0,ut0,r,rtol,atol))
    setappdata(pde_fig,'ncafd',ones(1,4))

    solveparams=str2mat('0','1000','10','pdeadworst','0.5','longest',...
        '0','1E-4','','fixed','Inf');
```




```

setappdata(pde_fig,'solveparam',solveparams)
% 对窗口进行配置属性
set(pde_fig,...
    'Tag','PDETool',...
    'NumberTitle','off',...
    'MenuBar','none',...
    'Units','pixels',...
    'MenuBar','none',...
    'Name','Drag-Free Satellite Simulation System')
axwidth = 0.8*Figpos(3);
axheight= axwidth/1.5/Figpos(4);
axstdpos = [0.1 0.12 0.8 0.8];
% 建立窗口中图形组件
ax=axes(...
    'Parent',pde_fig,...
    'Position',axstdpos,...
    'Units','normalized',...
    'DataAspectRatio',[1 1 1],...
    'Tag','PDEAxes');
view(ax,3) % 设定视角
% 建立菜单
[lbl,acc]=menulabel('&File');
file_menu = uimenu(pde_fig,'Accelerator',acc,'Label',lbl,...
    'Tag','PDEFileMenu','UserData',flags);
[lbl,acc]=menulabel('&New ^n');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,...
    'Interruptible','on','CallBack','pdetool(''new'')');
[lbl,acc]=menulabel('&Open... ^o');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,...
    'Interruptible','on','CallBack','pdetool(''open'')');
[lbl,acc]=menulabel('&Save ^s');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,...
    'Tag','PDESave','Enable','off','CallBack','pdetool(''save'')');
[lbl,acc]=menulabel('Save &As...');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,...
    'CallBack','pdetool(''save_as''),');
[lbl,acc]=menulabel('&Print...');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,'Separator','on',...
    'CallBack','pdetool(''print'')');
[lbl,acc]=menulabel('E&xit ^w');
uimenu(file_menu,'Accelerator',acc,'Label',lbl,...
    'Interruptible','on','Separator','on','CallBack','pdetool(''exit'')
');

% Edit menu:
[lbl,acc]=menulabel('&Edit');
edit_menu = uimenu(pde_fig,'Accelerator',acc,'Label',lbl,...
    'Tag','PDEEditMenu','UserData',[]);

```



```
[lbl,acc]=menulabel('&Undo ^z');
uimenu(edit_menu,'Accelerator',acc,'Label',lbl,'Enable','off',...
    'Tag','PDEUndo','CallBack','pdetool(''undo'')');
[lbl,acc]=menulabel('Cu&t ^x');
uimenu(edit_menu,'Accelerator',acc,'Label',lbl,'Separator','on',...
    'Tag','PDECut','CallBack','pdetool(''cut'',1)');
[lbl,acc]=menulabel('&Copy ^c');
uimenu(edit_menu,'Accelerator',acc,'Label',lbl,'Tag','PDECopy',...
    'CallBack','pdetool(''copy'')');
[lbl,acc]=menulabel('&Paste... ^v');
uimenu(edit_menu,'Accelerator',acc,'Label',lbl,'Enable','off',...
    'Tag','PDEPaste','CallBack','pdepsdlg');
[lbl,acc]=menulabel('Clea&r ^r');
uimenu(edit_menu,'Accelerator',acc,'Label',lbl,'Enable','off',...
    'Separator','on','Tag','PDEClear','CallBack','pdetool(''clear'')');
[lbl,acc]=menulabel('Select &All ^a');
uimenu(edit_menu,'Label',lbl,'Accelerator',acc,'Separator','on',...
    'Tag','PDESelall','CallBack','pdeselect(''select'',1)');

[lbl,acc]=menulabel('D&raw');
draw_menu = uimenu(pde_fig,'Accelerator',acc,'Label',lbl,...
    'Tag','PDEDrawMenu');

[lbl,acc]=menulabel('&Draw Mode');
uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDEDrawMode','Interruptible','on',...
    'CallBack','pdetool(''changemode'',1)');
[lbl,acc]=menulabel('&Rectangle/square');
draw_hndl(1)=uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDERect','Interruptible','on',...
    'CallBack','pdetool(''drawrect'',1)');
[lbl,acc]=menulabel('Rectangle/&square (centered)');
draw_hndl(2)=uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDERectc','Interruptible','on',...
    'CallBack','pdetool(''drawrect'',2)');
[lbl,acc]=menulabel('&Ellipse/circle');
draw_hndl(3)=uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDEEllip','Interruptible','on',...
    'CallBack','pdetool(''drawellipse'',1)');
[lbl,acc]=menulabel('Ellipse/&circle (centered)');
draw_hndl(4)=uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDEEllipc','Interruptible','on',...
    'CallBack','pdetool(''drawellipse'',2)');
[lbl,acc]=menulabel('&Polygon');
draw_hndl(5)=uimenu(draw_menu,'Label',lbl,'Accelerator',acc,...
    'Tag','PDEPoly','Interruptible','on',...
    'CallBack','pdetool drawline');
```



```
[lbl,acc]=menulabel('R&otate...');
uimenu(draw_menu,'Label',lbl,'Accelerator',acc,'Separator','on',...
    'Tag','PDERotate','CallBack','pdertdlg');

set(draw_menu,'UserData',draw_hndl);

[lbl,acc]=menulabel('E&xpport  Geometry  Description,  Set  Formula,
Labels...');
uimenu(draw_menu,'Label',lbl,'Accelerator',acc,'Tag','PDEExpGD',...
'Separator','on','CallBack','pdetool(''export'',1),'Enable','off');

[lbl,acc]=menulabel('P&DE');
pde_menu=uimenu(pde_fig,'Label',lbl,'Accelerator',acc,...
    'Tag','PDEPDEMenu','UserData',params);
[lbl,acc]=menulabel('&PDE Mode');
uimenu(pde_menu,'Label',lbl,'Accelerator',acc,'Tag','PDEMode',...
    'CallBack','pdetool(''pdemode'')');
[lbl,acc]=menulabel('&Show Subdomain Labels');

uimenu(pde_menu,'Label',lbl,'Accelerator',acc,'Tag','PDEShowPDESub',...
    'CallBack','pdetool(''pdesublbl'')');
[lbl,acc]=menulabel('P&DE Specification...');
uimenu(pde_menu,'Label',lbl,'Accelerator',acc,'Tag','PDEParam',...
    'CallBack','pdetool(''set_param'')');
[lbl,acc]=menulabel('E&xpport PDE Coefficients...');

uimenu(pde_menu,'Label',lbl,'Accelerator',acc,'Tag','PDEExpParam',...
    'Separator','on','CallBack','pdetool(''export'',4)');

% Window menu:
[lbl,acc]=menulabel('&Window');
uimenu(pde_fig,'Label',lbl,'Accelerator',acc,...
    'CallBack','winmenu; drawnow',...
    'Tag','winmenu');

end
```

基于该程序得到卫星三维建模与有限元分析工具。该工具初始化界面如图 16-1 所示；图 16-2 为该工具中建模的基本单位，由图中可以看出，基本单位包括球体、圆柱体和长方体等；图 16-3 为基于该工具建立的一个简易卫星模型；图 16-4 为针对卫星模型进行有限元划分后的图形。

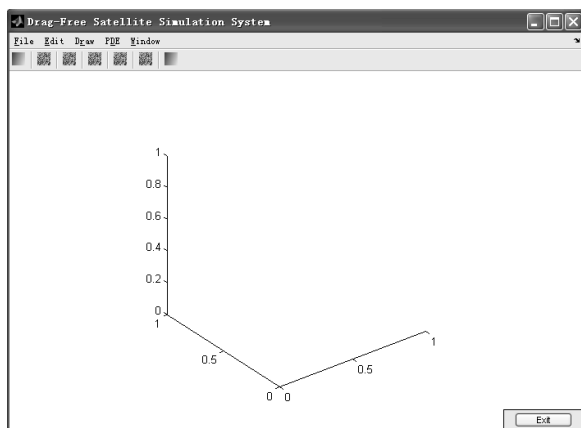


图 16-1 卫星三维建模与有限元分析工具初始化界面

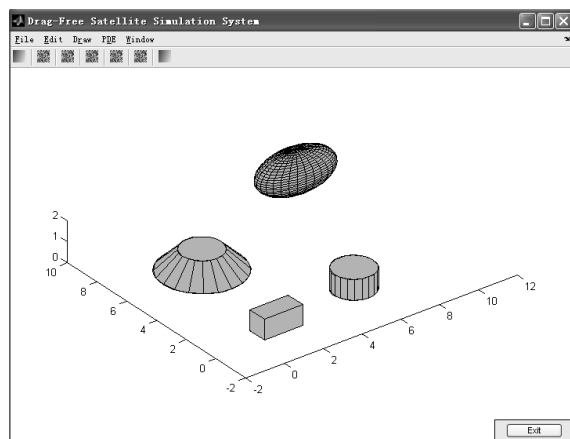


图 16-2 卫星三维建模与有限元分析工具建模基本单位

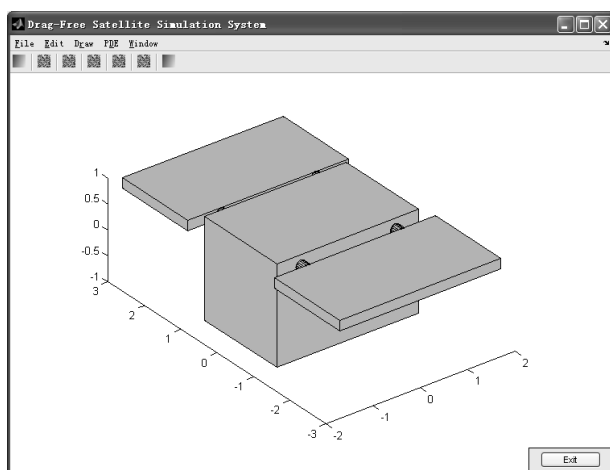


图 16-3 卫星实例

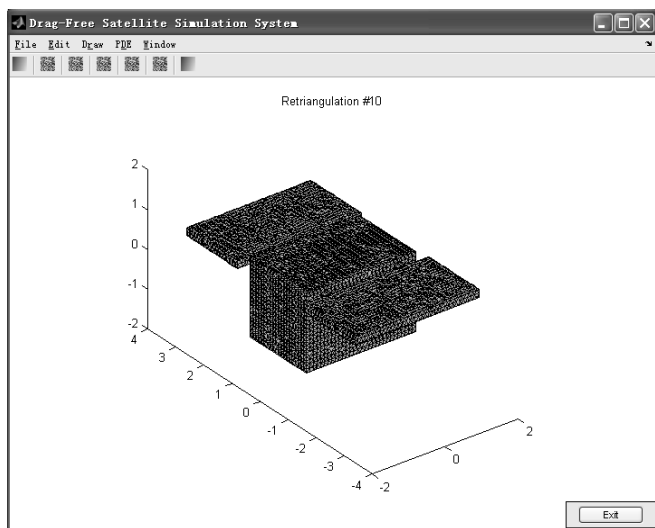


图 16-4 有限元划分后图形

16.3 本例小结

本例介绍了基于 M 语言建立 GUI 界面的过程，对如何建立界面的菜单、工具栏以及建立三维图形等进行详细的介绍，并通过建立卫星三维模型并进行有限元分析为例来检验其能够实现的功能。

第五部分 热工程仿真实例



引言——Simscape 语言简介（上）

Simscape 语言通过使用户可以自定义在模型库中不存在的模型，从而扩展了 Simscape 建模能力。Simscape 语言专用于 Simscape 建模，具有以下两个特点：

- 基于 MATLAB 建模语言。
- 用于多物理对象建模。

Simscape 语言，使物理系统的建模更容易，更直观。它可以让用户通过文本文件自定义模型元件，该元件包含完整的参数，物理连接并用表示因果性的隐式差分代数方程模块（DAE）来表示。用户所创建的组件可以重用 Simscape 的域定义，以确保用户自定义的组件兼容标准的 Simscape 组件。用户也可以定义自己的物理域，并可以自动建立和管理 Simscape 组成部分的功能块库，使能够在用户所在组织中共享这些模型。

1. Simscape 语言例子——线性电阻

通过一个 Simscape 语言例子来介绍 Simscape 语言的构成特点，该例子目的是建立一个简单的线性电阻。

线性电阻器是一个简单的电气部件，其等式描述如式（E1）所示：

$$R = \frac{V}{I} \quad (\text{E1})$$

式中 R 表示电阻， V 表示电压， I 表示电流。

下面的程序给出该电阻

```
component my_resistor
% linear resistor
% 电压、电流和电阻之间的关系为 V=I*R
% R 为常值电阻，
%
% 电阻正负极用+和-号表示

nodes
    p = foundation.electrical.electrical; % +:left
```



```

    n = foundation.electrical.electrical; % -:right
end
variables
    i = { 0, 'A' };
    v = { 0, 'V' };
end
parameters
    R = { 1, 'Ohm' }; % Resistance
end

function setup
    across( v, p.v, n.v );
    through( i, p.i, n.i );
    if R <= 0
        error( 'Resistance value must be greater than zero' );
    end
end

equations
    v == i*R;
end

end

```

文件中第一行“component”表示其是一个部件模型文件，文件名为“my_resistor”。文件中第一行下面的一部分为注释部分。用于说明该文件描述对象的一些基本特性。

注释下面的为声明部分，在声明部分共对两个节点、两个变量和一个参数进行了声明。其中两个节点声明是基于 Simscape 基础工具箱中定义的域模型进行的，“foundation”表示 Simscape 基础工具箱底层域，“electrical”是基于“foundation”关于电子对象的子域，基于“foundation”的其他子域还有流体、机械和传动等。这里声明使用“electrical”子域，表明该线性电阻能够和 Simscape 基础工具箱中电子部分的模块进行连接仿真。此外，从文件还可以看出声明时变量是以数值和单位组形式出现的。例如对变量 i 的声明，中括号中 0 表示其数值，“A”表示其单位。

接下来是初始化部分，初始化部分始于“function setup”，结束于与之对应的“end”。初始化部分包含一个 across 函数、一个 through 函数和一个错误提示。其中错误提示如果设置的 R 值小于零，则终止程序，并在命令行窗口提示错误。下面对 across 函数和 through 函数进行介绍，这两个函数在 Simscape 语言中经常使用到。

首先是 across 函数，其调用格式如下：

```
across(variable1, node1.variableA, node2.variableB)
```

其中 variable1 是声明中的变量，node1 表示节点 1，node2 表示节点 2，variableA 是节点 1 中的变量，variableB 是节点 2 中的变量。

使用 across 函数需要注意以下三方面：



- 函数输入的单位必须一致。
- 第二个输入变量和第三个输入变量没有必要是同一个域模型，例如第二个输入变量可以是一相电机电压，第三个输入变量可以是三相电机电压。
- 第二个输入变量和第三个输入变量中可以有一个为“[]”，此时作为参考节点。

文件中 `across(v, p.v, n.v)` 的作用是将变量 `v` 定义为从节点 `p` 到节点 `n` 的势变量。

其次是 `through` 函数，与 `across` 函数类似，其调用格式如下：

```
through(variableI, node1.variableA, node2.variableB)
```

其中 `variableI` 是声明中的变量，`node1` 表示节点 1，`node2` 表示节点 2，`variableA` 是节点 1 中的变量，`variableB` 是节点 2 中的变量。

使用 `through` 函数与使用 `across` 函数类似，也需要注意单位等三个方面。

文中 `through(i, p.i, n.i)` 的作用是将变量 `i` 定义为从节点 `p` 到节点 `n` 的流变量。实际计算时满足如式 (E2) 所示关系：

$$i = p.i - n.i \quad (E2)$$

文件中最后一部分是公式部分，始于“`equations`”，结束于“`end`”。公式部分中“`v == i*R`”建立了简单电阻中电压电流和电阻的关系。

将上述文件名保存为“`my_resistor.ssc`”，如果采用默认模块，其外形如图 E1 所示。



图 E1 my_resistor 外形图

2. Simscape 文件

Simscape 文件是 MATLAB 环境中特定的文件格式，其后缀名为“`.ssc`”。

Simscape 文件用于对物理对象进行建模，使用的语言与 MATLAB 语言存在不同，但是与 MATLAB 底层语言存在交互。

Simscape 文件必须存放于以“+”号开头的文件名目录或多层目录下，如：

- `/+MyPackage/MyComponent.ssc`。
- `/+MyPackage/+Subpackage/.../MyComponent.ss`。

Simscape 文件按照类别可分为域模型文件和部件模型文件两种，这两种文件分别对应两种模型：

- 域模型。域模型描述了某一特定物理域中能量与数据的交换。这些域可以指任意物理领域，比如电学领域、热学领域、流体领域等。通过定义物理域中特定的势变量和流变量，从而反映特定物理域的特性。势变量是指随着距离变化的量，而流变量在未有新的节点加入时，并不随距离变化。势变量如电压，流变量如电流。
- 部件模型。在建立域模型之后，可进一步建立部件模型，部件模型表示特定的物理部件，如电容、电阻和电源等。部件模型中使用到的势变量和流变量必须和域模型中的变量定义一致，除域模型中的势变量和流变量外，部件模型中还可以定义其他参数和变量，以进一步完善部件模型特性。

举个例子，为建立一个不同于 Simscape 基础工具箱的新的流体模型，则可将其取名



为 MyVarOrifice.ssc，并基于 Simscape 基础工具箱中流体域模型进一步建立模型。但是如果要建立一个新域中的部件模型，则首先需要建立域模型，然后在域模型的基础上建立部件模型。例如要建立一个热流部件模型，则可以先建立热流域模型文件 thermohydraulic.ssc，该热流域模型表征的是流体的温度，然后在该域模型基础上，可进一步建立特定的部件模型文件。

3. 文件结构

上面介绍了域模型文件和部件模型文件，下面对这两类文件结构分别予以介绍。

(1) 域模型文件格式，域模型文件分为两部分：文件类型和文件名部分以及声明部分。

文件类型和文件名部分采用如下的格式：

```
文件类型 文件名
```

其中文件类型表示该文件是域模型文件或部件模型文件，如果表示域模型文件，则采用“domain”，如果表示部件模型文件，则采用“component”，文件类型及文件名部分例子如下所示：

```
domain rotational
```

其中“domain”表示该文件为域模型文件，而“rotational”表示机械系统中的旋转域。

```
domain spring
```

其中“domain”表示该文件为域模型文件，而“spring”表示机械系统中弹性域。域模型文件中声明部分在后续内容中详细介绍。

(2) 部件模型文件，部件模型文件分为四部分：文件类型和文件名部分、声明部分、初始化部分和方程部分。

其中文件类型和文件名部分与域模型文件中类似，只是将“domain”换成“component”。部件模型文件中文件类型和文件名部分的例子如下：

```
component capacitance
```

其中“component”表示该文件为部件模型文件，“capacitance”表示该文件为电容文件。

```
component resistance
```

其中“component”表示该文件为部件模型文件，“resistance”表示该文件为电阻文件。

部件模型文件中声明部分、初始化部分和方程部分将在后续内容中详细介绍。

4. 文件内容

从 Simscape 文件内容来分，也可以分为两部分：接口声明部分以及实现部分。



其中接口声明部分对于域模型指定势变量和流变量以及参数；而实现部分只存在于部件模型文件中，包含初始化和等式两部分，其中初始化部分用于定义部件模型文件中参数的初始值，而等式部分用于定义部件中变量及输入输出间的关系。

通过将 Simscape 文件与 MATLAB 中的类进行对比可以看出，两者既存在相似，也存在不同。其中相似部分为 Simscape 文件中接口声明部分以及实现部分与类中的构造对象及函数对象都作为类的具体实现。而 Simscape 文件与 MATLAB 中的类间不同点有两个。

(1) Simscape 文件并不将参数直接传递给其函数，这减少了语法的复杂性，并不失一般性。

(2) Simscape 文件中函数在部件模型文件生命周期中起特定的作用，具体的作用如图 E2 所示。

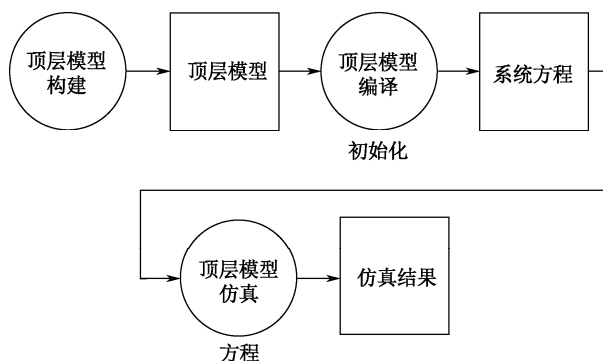


图 E2 Simscape 文件中函数在生命周期中作用

由图可以看出，Simscape 文件生命周期分为三部分：第一部分是顶层模型构建，通过构建得到顶层模型；第二部分是顶层模型编译，通过初始化得到系统方程；第三部分是顶层模型仿真，基于方程进行仿真得到仿真结果。

其中顶层模型构建分为四步。

- (1) 激活文件名。
- (2) 为顶层模型添加函数。
- (3) 准备参数。
- (4) 将不同部件模型连接起来从而构建成一个完整的物理对象顶层模型。

顶层模型编译通过调用初始化函数进行完成，而顶层模型仿真则是基于方程函数完成并最终得到仿真结果。

5. 生成新的物理域

物理域是能够通过事先定义好的势变量和流变量连接不同部件从而组成物理网络的物理仿真环境。由某一域模型生成的部件模型只能与同一域模型生成的部件模型进行连接。

如果用户所需的自定义部件模型所属的域模型是 Simscape 基础工具箱中已经存在的，则并不需要定义新的域模型。Simscape 基础工具箱中所有的域模型有机械平动、机



械转动、电子和流体等。这些域模型是 Simscape 基础工具箱中所有部件模型的基础，Simscape 基础工具箱中部件模型的势变量和流变量必须与这些域模型中的定义一致。如果用户所需的自定义部件模型也采用 Simscape 中的域模型，则其中的势变量和流变量也必须与这些域模型中的定义一致。

如果用户所需的自定义部件模型所属的域模型是 Simscape 基础工具箱中并不存在，则需要定义新的域模型。比如前面介绍的需要建立热流域中的部件模型，则就需要首先建立满足需要的域模型，然后再基于该模型建立特定的部件模型。

注意：基于自定义域模型建立的部件与 Simscape 基础工具箱中的部件不能交互。

前面介绍了域模型通过域模型文件进行定义，而域模型文件分为类型和文件名以及声明两部分，类型和文件名部分在前面已经定义过，这里对声明部分予以介绍。

首先给出一个域模型文件实例：

```
domain t_hyd
variables
  p = { 1e6, 'Pa' }; % pressure
end
variables(Balancing = true)
  q = {1e-3, 'm^3/s'}; % flow rate
end
parameters
  t = {303, 'K'}; % flow temperature
end
end
```

由域模型文件实例可以看出声明部分分为势变量声明、流变量声明和参数定义三部分，其中参数定义部分是可选的。

势变量声明始于“variables”，结束于“end”，其中包含所有势变量的声明，本例中只有一个势变量声明“p = { 1e6, 'Pa' }”，其中“p”表示变量名，“1e6”表示该变量的值，“Pa”表示该变量的单位。

注意：域模型声明中变量需要以数值和单位组形式给出。且注释部分具有特定的作用，该作用在后面会继续介绍。

流变量声明始于“variables(Balancing = true)”，结束于“end”。参数声明部分始于“parameters”，结束于“end”。与势变量声明类似，所有的流变量声明和参数声明都集中在了一起。

6. 部件模型

在物理建模中，有两种建模方式。

(1) 数学方程式。该种建模方式基于物理对象特性，并利用数学方程将物理对象特性描述出来，再利用部件模型文件完成部件建模。

(2) 构建式。该种建模方式通过多个子功能块完成物理对象建模。每个子功能块对应部分功能特性，这要求首先将物理对象功能分解为几个子功能块。



前面介绍了部件模型文件分为四部分并对第一部分内容予以介绍，下面对后面三部分予以介绍。首先给出部件模型文件实例如下：

```
component h_temp_ref
% temperature reference
nodes
    a = THyd.t_hyd; % t_hyd node
end
variables
    q = { 0, 'm^3/s' };
end
function setup
    through( q, a.q, [] );
end
equations
    a.p == 0;
end
end
```

该文件实例中第一行中“component”表示是部件模型文件，“h_temp_ref”表示文件名。

该文件的声明部分包含两个部分：第一部分始于“nodes”，结束于“end”；第二部分始于“variables”，结束于“end”。声明部分除“nodes”和“variables”两类外，还存在“parameters”、“inputs”和“outputs”其他三类。下面将对这几类声明进行介绍。

文件的初始化部分始于“function setup”，结束于“end”。

文件的方程部分始于“equations”，结束于“end”。



第 17 例 车载 stewart 平台建模与仿真



本例在介绍 Simscape 语言的基础上,以 stewart 平台为对象,介绍建立机械对象并仿真的过程。通过本例学习,需要掌握以下两点:

- 基于 Simscape 外形结构与定制方法。
- 基于 Simscape 工具箱的车载 stewart 平台动力学建模与仿真。

17.1 Simscape 语言简介 (中)

该部分主要介绍基于 Simscape 生成模块的外形结构与定制方法。Simscape 生成模块如果未做修改时,具有默认的一些特性,下面首先介绍 Simscape 默认模块外形。

当建立定制模块时,模块的名称和参数是从部件文件中得到的,默认模块图标是由一个矩形以及模块名称、端口名称组成的。

下面给出一个部件文件实例以及对应生成的默认模块外形。其中文件如下:

```
component spring
  nodes
    r = foundation.mechanical.rotational.rotational;
    c = foundation.mechanical.rotational.rotational;
  end
  parameters
    k = { 10, 'N*m/rad' };
  end
  variables
    theta = { 0, 'rad' };
    t = { 0, 'N*m' };
    w = { 0, 'rad/s' };
  end
  function setup
    if k < 0
      error( 'Spring rate must be greater than zero' );
    end
    through( t, r.t, c.t );
    across( w, r.w, c.w );
  end
  equations
```



```
t == k * theta;  
w == theta.der;  
end  
end
```

而对应的模块外形如图 17-1 所示。

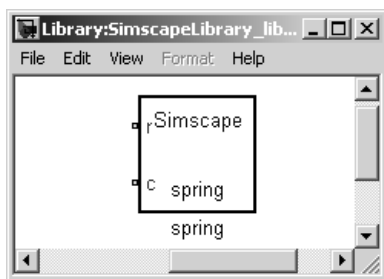


图 17-1 默认模块外形图示例

由模块的默认图示例可以看出其由矩形框部分以及模块名称两部分组成，其中模块名称“spring”与模型文件中名次一致，矩形框部分中包含三部分：蓝色字体的“Simscape”表明这是基于 Simscape 语言编写的模块；红色字体的“r”和“c”与模型文件中声明的两个节点一致，黑色字体的“spring”也与模型文件中名称一致。

双击模块，打开属性对话框如图 17-2 所示，对话框分为三部分：第一部分为文件说明部分，这里未予说明，只给出默认的源文件链接“View source for spring”；第二部分为参数设置部分。因为源文件只给出一个参数 k ，所以这里也只显示出对应 k 的设置选项，共有两个，一个是值，默认为文件中给出的 10，另一个是单位，默认值也采用文件中给出的“N*m/rad”；第三部分是“确定”、“取消”、“帮助”和“应用”等按钮。

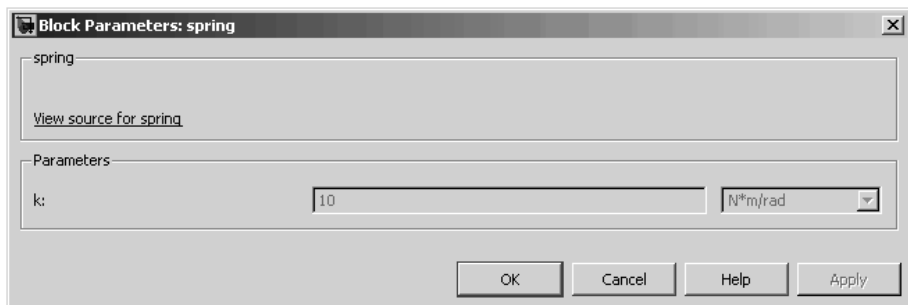


图 17-2 模块双击打开属性对话框

以上给出的是模块默认的外形，用户也可以根据需要对模型的外形进行定制从而达到更加直观、易于理解的目的。

1. 定制模块名称

前面介绍到，在默认情况下，模块名称与文件名一致，如果用户需要对模块名称进行定制，可以通过在部件文件中文件名下面一行进行注释得到。注释的例子如下所示：



```
component spring
%Rotational Spring
...
end
```

上述文件中注释部分“Rotational Spring”即为定制模型文件名称。
此时生成的模块外观如图 17-3 所示。

2. 定制模块属性对话框

属性对话框有三部分内容,其中前两部分内容是可以定制的,下面分别介绍这两部分内容定制的方法。

首先是模块说明部分。前面介绍到默认情况下说明部分只有源文件链接,为对模型进一步予以说明,可以通过在模型名称注释部分下面添加模型说明部分的注释。下面给出一个注释的例子:

```
component spring
%Rotational Spring
% This block implements a simple rotational spring
...
end
```

该例子对应的模型属性对话框如图 17-4 所示。

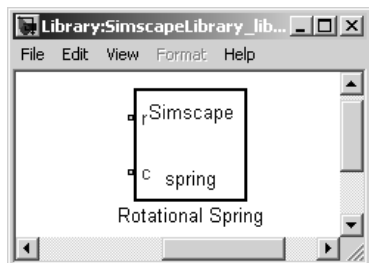


图 17-3 模型名称定制结果图

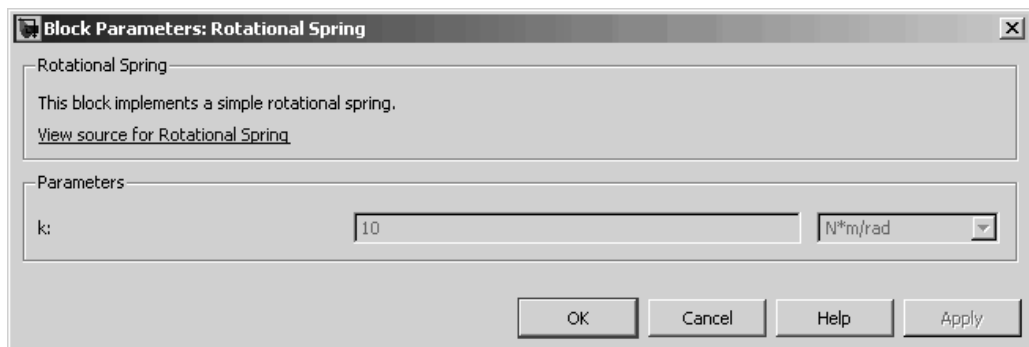


图 17-4 模型说明定制示例

由图可以看出添加的模型说明部分“This block implements a simple rotational spring”与文中注释部分相同。

说明:如果在说明部分需要分段说明,则可以在段落之间添加一个空白说明符,如下例所示。

```
% end of one paragraph
%
% beginning of the next paragraph
```



下面介绍如何对参数设置部分进行定制。

如果需要定制参数说明部分，则可以在参数声明部分的同一行后面添加注释。如下例所示：

```
component spring
%Rotational Spring
% This block implements a simple rotational spring
...
parameters
    k = { 10, 'N*m/rad'}; % Spring rate
end
...
end
```

对应该例的属性对话框如图 17-5 所示。

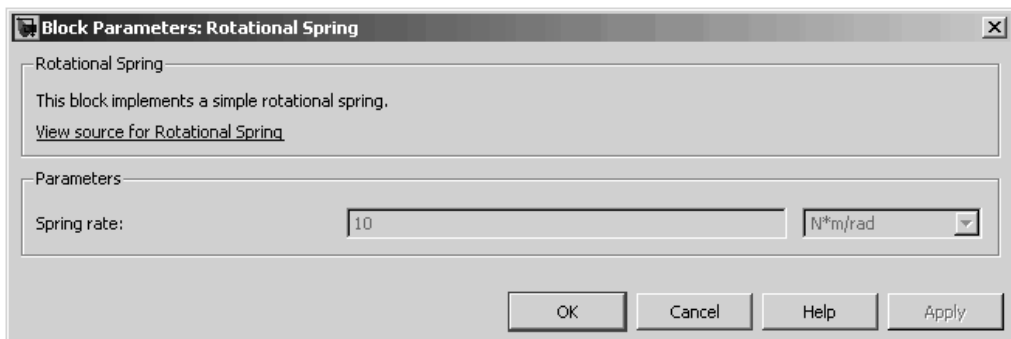


图 17-5 定制参数说明示例

3. 定制模块端口的名称及位置

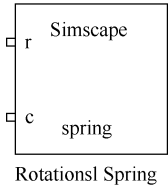
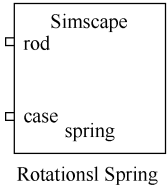
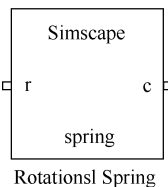
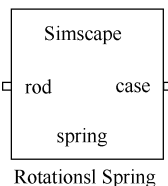
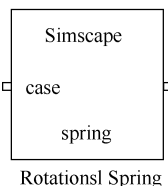
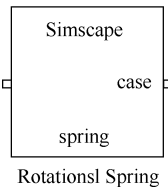
默认模块端口都处于矩形框的左边且等距分布，这在实际使用时存在两个缺点：一是如果端口较多，则端口之间排布紧密，不利于连线；二是输入输出都集中于一端，不符合通常的使用习惯。

为使模块的端口更便于使用，可对其进行定制。定制的方法与前面类似，可以通过在节点、输入和输出变量等声明同一行后面给出定制注释。

由于端口包括位置和名称，因此定制时也分为两类：一类是对端口位置进行定制，可将任一端口置于上下左右等位置；另一类是对每一端口的名称进行修改。下面通过实例予以说明。表 17-1 列出了几个对端口进行定制的实例。

由表可以看出，端口自定义注释语法主要分为两部分，两部分由冒号连接，前一部分为端口名称，后一部分为端口位置，其中端口位置可为“left”、“right”、“top”和“bottom”，分别对应左、右、上和下的位置。

表 17-1 端口定制实例

语 法	模 块 外 形
<pre>nodes r = foundation.mechanical.rotational.rotational; c = foundation.mechanical.rotational.rotational; end</pre>	 Rotationsl Spring
<pre>nodes r = foundation.mechanical.rotational.rotational; % rod c = foundation.mechanical.rotational.rotational; % case end</pre>	 Rotationsl Spring
<pre>nodes r = foundation.mechanical.rotational.rotational; c = foundation.mechanical.rotational.rotational; % c:right end</pre>	 Rotationsl Spring
<pre>nodes r = foundation.mechanical.rotational.rotational; % rod c = foundation.mechanical.rotational.rotational; % case:right end</pre>	 Rotationsl Spring
<pre>nodes r = foundation.mechanical.rotational.rotational; % case c = foundation.mechanical.rotational.rotational; % :right end</pre>	 Rotationsl Spring
<pre>nodes r = foundation.mechanical.rotational.rotational; % c = foundation.mechanical.rotational.rotational; % case:right end</pre>	 Rotationsl Spring



4. 定制模块图标的大小

除了对模块外观在给定的框架下进行定制外，用户还可以用制作好的图片来表示模块，制作的图片具有更加形象直观的效果。

为使用用户自己的图片，需要完成两部分工作：首先准备好一个与部件文件名相同名称的图片并置于同一目录下，图片的格式保存为“JPG”、“BMP”和“PNG”这三种中的任意一种；其次按照特定的语法结构进行注释。

注释的例子如下：

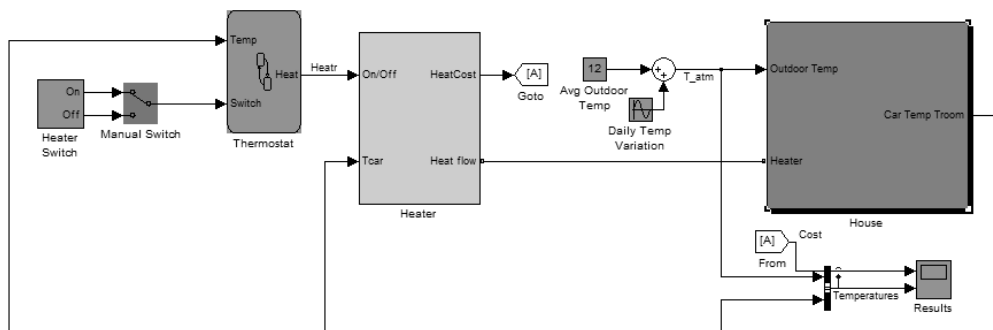
```
component spring
% Rotational Spring : 0.5 : fixed
```

例子中注释分为三部分：第一部分与前面一样，定义模型名称；第二部分是“0.5”，假设用户准备的图片像素为 80×120 ，则 MATLAB 首先将其像素较小的一边缩减为 40，此时图片缩减为 40×60 ，然后与该部分的 0.5 相乘，得到最终表现的像素为 20×30 ，默认值为 1；第三部分为“fixed”，该部分对应的值有两个，为“rotation”和“fixed”，其中 rotation 表示当对模块进行旋转操作时，图片也随之旋转，而 fixed 则表示当模块进行旋转操作时，图片不随之旋转，该部分的默认值是“rotation”。

说明：当存在相同名称图片但没有注释语言时，则采用对应的默认值，即模块名称采用文件名，缩放比例为 1，是否旋转属性选择旋转。

17.2 汽车温度调节系统仿真

基于 Simscape 建立汽车温度调节系统如图 17-6 所示。由图可知系统分为以下几部分。



Car Heating System

The demo represents a simple car heating system consisting of a heater, thermostat, and a car structure with four thermally distinguishable parts: inside air, house walls, windows, and roof. The car exchanges heat with the environment through its walls, windows, and roof. Each path is simulated as a combination of a thermal convection, thermal conduction, and the thermal mass. The heater starts pumping hot air if room temperature falls below 18 deg C and is turned off if temperature exceeds 23 deg C. As a result, you can monitor the temperatures of house parts and heat losses through them. The manual switch allows you to investigate system behavior with the heating system turned off.

图 17-6 汽车温度调节系统

(1) 温度动力学部分。该部分包含基于 Simscape 生成的温度动力学模块，具体如图 17-7



所示。其中包含温度敏感器（如图 17-8 所示）、温度参考源和温度传播模块等。

（2）温度调节开关，如图 17-9 所示。当开关打开时，开始调节温度，关闭时，则不予调节。

（3）温度控制器，如图 17-10 所示。当开关打开时，温度调节按照该控制器给出的指令进行调节。

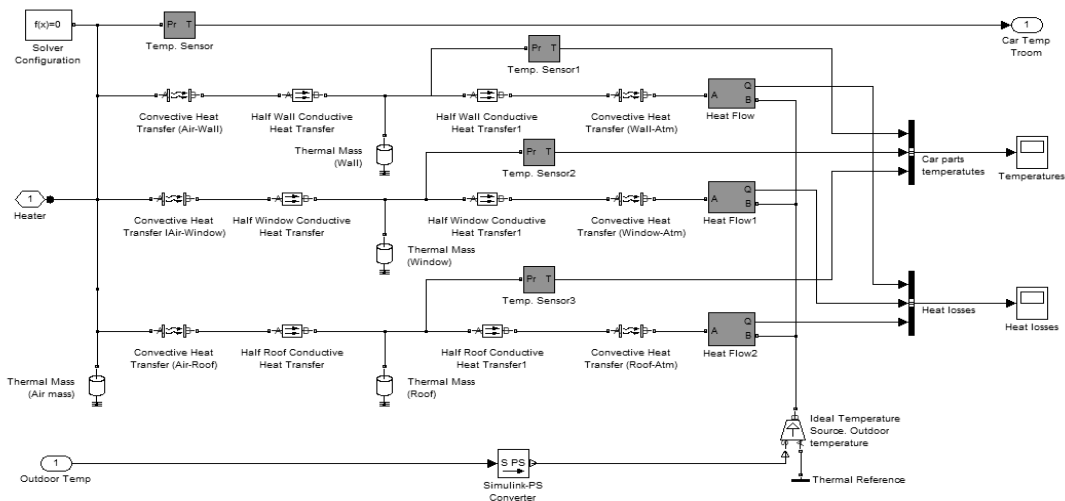


图 17-7 汽车温度系统动力学



图 17-8 温度敏感器模块

图 17-9 温度调节器开关

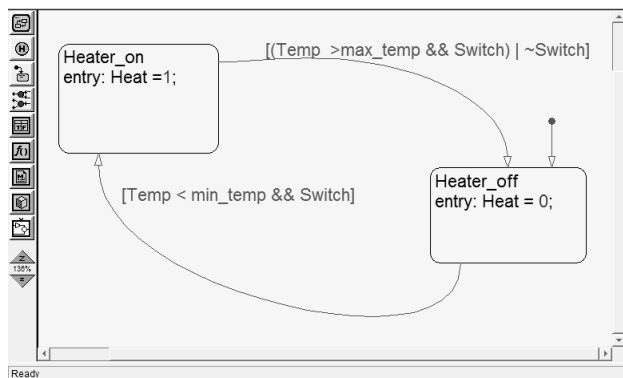
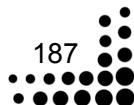


图 17-10 温度控制器





(4) 温度指令生成部分,如图 17-11 所示。温度开关指令及温度控制器指令都输入到该部分,并引入滤波环节最终生成控制温度变化的指令。

仿真结果如图 17-12 和图 17-13 所示。可以看出汽车温度在一定幅度内进行变化,表明仿真系统能够对汽车温度调节进行模拟仿真。

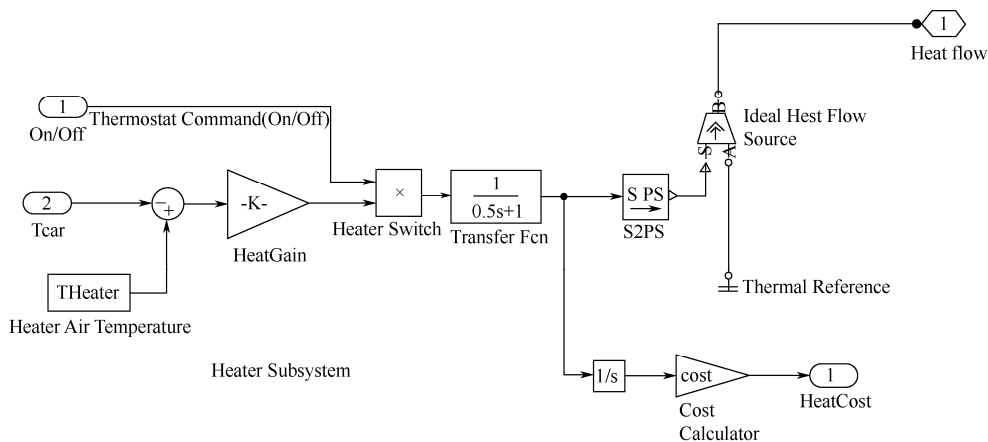


图 17-11 温度指令生成部分

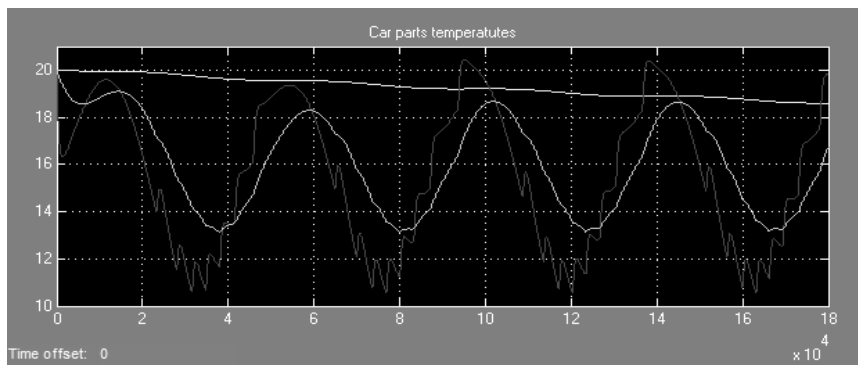


图 17-12 温度调节仿真结果 1

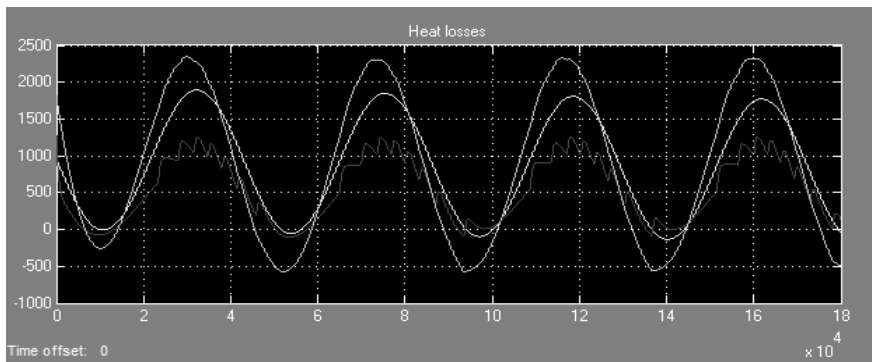


图 17-13 温度调节仿真结果 2



温度敏感器模块代码如下：

```
component temperature
% Ideal Temperature Sensor :0.8
% The block represents an ideal temperature sensor, that is, a device
% that determines the temperature differential measured between two points
% without drawing any heat. The temperature differential, T, is returned
% at the physical signal port T. Connections A and B are conserving thermal
% ports.
%
% The sensor is oriented from A to B and the measured temperature is
% determined as  $T = T_A - T_B$ .

% Copyright 2005-2008 The MathWorks, Inc.

nodes
    A = foundation.thermal.thermal; % A:left
    B = foundation.thermal.thermal; % B:right
end

outputs
    T = { 0, 'K' }; % T:right
end

variables
    Qs = { 0, 'J/s' };
    Ts = { 0, 'K' };
end

function setup
    through( Qs, A.Q, B.Q );
    across( Ts, A.T, B.T );
end

equations
    Qs == 0;
    Ts == T;
end

end
```

17.3 本例小结

本例介绍了 Simscape 生成模块的外形结构与定制方法，建立汽车温度调节系统模型并进行仿真，仿真结果表明基于 Simscape 语言能够对多物理系统进行建模仿真，通过本例学习应进一步熟悉 Simscape 语言结构及使用方法。



第 18 例 车载简易温度检测系统

上一例介绍了基于 Simscape 语言进行车载温度系统的建模与仿真过程，其中仿真的模块采用的是 Foundation 中自带的模块，因此本例进一步介绍如何基于 Simscape 语言创建新的域模型和部件模型，并通过实例进行演示。通过本例学习需要掌握以下两点：

- 基于 Simscape 语言创建新域模型过程。
- 基于新的域模型创建部件模型过程。

18.1 Simscape 语言简介（下）

Simscape 工具箱自身带了已经建立好的域模型，为减小工作量，提高开发效率，用户可以在此基础上开发自己的模型。

1. 电子域（Electrical Domain）

域模型声明如下：

```
domain electrical
% Electrical Domain

% Copyright 2005-2008 The MathWorks, Inc.

parameters
    Temperature = { 300.15 , 'K' }; % Circuit temperature
    GMIN        = { 1e-12 , '1/Ohm' }; % Minimum conductance, GMIN
end

variables
    v = { 0 , 'V' };
end

variables(Balancing = true)
    i = { 0 , 'A' };
end

end
```



为引用该域模型，可利用下述语句：

```
foundation.electrical.electrical
```

2. 流体域 (Hydraulic Domain)

域模型声明如下：

```
domain hydraulic
% Hydraulic Domain

parameters
    density      = { 850 , 'kg/m^3' }; % Fluid density
    viscosity_kin = { 18e-6 , 'm^2/s' }; % Kinematic viscosity
    bulk         = { 0.8e9 , 'Pa' }; % Bulk modulus at atm. pressure
and no gas
    alpha        = { 0.005 , '1' }; % Relative amount of trapped air
end

variables
    p = { 0 , 'Pa' };
end

variables(Balancing = true)
    q = { 0 , 'm^3/s' };
end

end
```

为引用该域模型，可利用下述语句：

```
foundation.hydraulic.hydraulic
```

3. 电磁域 (Magnetic Domain)

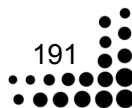
域模型声明如下：

```
domain magnetic
% Magnetic Domain

parameters
    mu0 = { 4*pi*1e-7 'Wb/(m*A)' }; % Permeability constant
end

variables
    mmf = { 0 , 'A' };
end

variables(Balancing = true)
    phi = { 0 , 'Wb' };
end
```





```
end
```

```
end
```

为引用该域模型，可利用下述语句：

```
foundation.magnetic.magnetic
```

4. 机械转动域 (Mechanical Rotational)

域模型声明如下：

```
domain rotational
% Mechanical Rotational Domain
variables
    w = { 0 , 'rad/s' };
end

variables(Balancing = true)
    t = { 0 , 'N*m' };
end

end
```

为引用该域模型，可利用下述语句：

```
foundation.mechanical.rotational.rotational
```

5. 机械平动域 (Mechanical Translation)

域模型声明如下：

```
domain translational
% Mechanical Translational Domain

variables
    v = { 0 , 'm/s' };
end

variables(Balancing = true)
    f = { 0 , 'N' };
end

end
```

为引用该域模型，可利用下述语句：

```
foundation.mechanical.translational.translational
```




6. 热域 (Thermal Domain)

域模型声明如下：

```
domain thermal
% Thermal domain

variables
    T = { 0 , 'K' };
end

variables(Balancing = true)
    Q = { 0 , 'J/s' };
end

end
```

为引用该域模型，可利用下述语句：

```
foundation.thermal.thermal
```

7. 气体域 (Pneumatic Domain)

域模型声明如下：

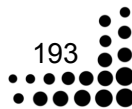
```
domain pneumatic
% Pneumatic 1-D Flow Domain

parameters
    gam = { 1.4, '1' };           % Ratio of specific heats
    c_p = { 1005 , 'J/kg/K' };    % Specific heat at constant pressure
    c_v = { 717.86 , 'J/kg/K' };  % Specific heat at constant volume
    R   = { 287.05, 'J/kg/K' };   % Specific gas constant
    viscosity = { 18.21e-6, 'Pa*s' }; % Viscosity
    Pa  = { 101325, 'Pa' };       % Ambient pressure
    Ta  = { 293.15, 'K' };       % Ambient temperature
end

variables
    p = { 0 , 'Pa' };
    T = { 0 , 'K' };
end

variables(Balancing = true)
    G = { 0 , 'kg/s' };
    Q = { 0 , 'J/s' };
end

end
```





为引用该域模型，可利用下述语句：

```
foundation.pneumatic.pneumatic
```

18.2 机载简易温度检测仿真

下面介绍如何基于 Simscape 语言创建一个简单的自定义工具箱。

该工具箱包含三个模块，分别是：流体温度模块、流体温度参考模块和流体温度敏感器模块。基于该工具箱可以实现简单的流体温度生成及敏感的功能。

(1) 首先建立一个文件夹，名称为“+THyd”，并将所有建立的文件都放在这个文件夹中，建完之后将该文件夹的路径添加到 MATLAB 所识别的路径中，或者将 MATLAB 当前工作路径设置为该文件夹路径。

(2) 创建域模型文件 t_hyd.ssc，其内容如下：

```
domain t_hyd
variables
    p = { 1e6, 'Pa'}; % pressure
end
variables(Balancing = true)
q = { 1e-3, 'm^3/s'}; % flow rate
end
parameters
t = { 303, 'K'}; % fluid temperature
end
end
```

(3) 创建部件文件 hyd_temp.ssc 如下：

```
component (Propagation = source) hyd_temp
% Hydraulic Temperature
% 为其他模块提供流体温度
parameters
t = { 303, 'K'}; % Fluid temperate
end
nodes
    a = THyd.t_hyd; % t_hyd node
end
function setup
    a.t = t; % 设置节点温度至指定值
end
end
```

(4) 创建部件文件 h_temp_sensor.ssc 如下：

```
component h_temp_sensor
% Hydraulic Temperature Sensor
```

```
% 用于测量其他模块温度
outputs
    T = { 0, 'K'}; % T:right
end
nodes
    a = THyd.t_hyd; % t_hyd node
end
equations
    T == a.t;
end
end
```

(5) 建立部件文件 h_temp_ref.ssc 如下：

```
component h_temp_ref
% Hydraulic Temperature Reference
% 为其他模块提供温度参考
nodes
    a = THyd.t_hyd; % t_hyd node
end
variables
    q = {0, 'm^3/s'};
end
function setup
    through(q, a.q, []);
end
equations
    a.p == 0;
end
end
```

(6) 将以上建立的几个文件置于同一以“+”开头的文件夹下，并把 MATLAB 当前工作路径设置为该文件夹路径。此时在 MATLAB 命令行中输入以下命令：

```
ssc_build THyd;
```

(7) 上述命令建立模型库并在同一文件夹下生成“THyd_lib.mdl”文件，为使用该模型，需要重启 MATLAB，并在命令行中输入：

```
THyd_lib
```

则打开建立的模型库如图 18-1 所示。

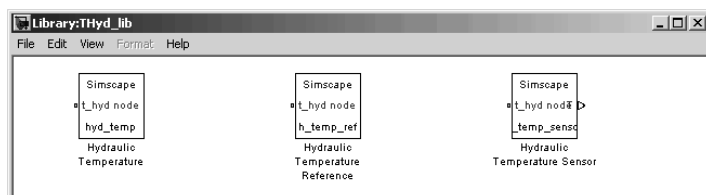


图 18-1 THyd_lib 模型库



(8) 在 MATLAB 命令行中输入：

```
ssc_new
```

则打开如图 18-2 所示部分定制模型文件。

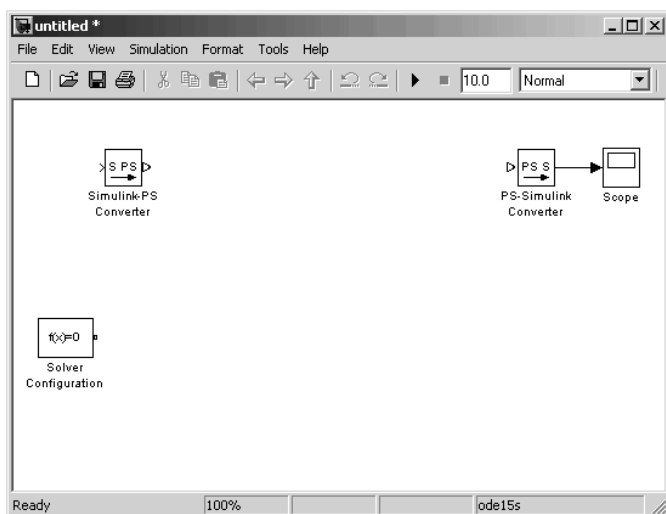


图 18-2 部分定制模型文件

(9) 删除“Simulink-PS Converter block”，因为建立的模型中不需要 Simulink 输入信号。

(10) 基于 THyd_lib 模型库建立如图 18-3 所示的温度检测模型。

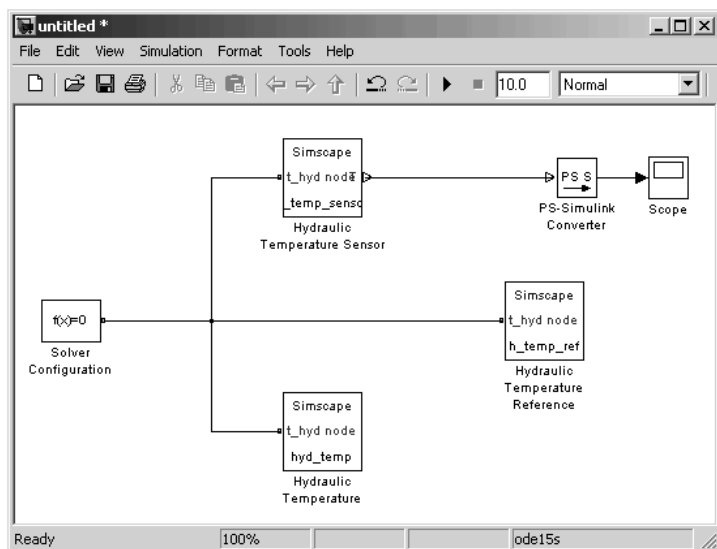


图 18-3 温度检测模型

(11) 仿真上述模型，则示波器的温度显示为 333K。

(12) 双击打开“Hydraulic Temperature block”，将“Fluid temperature”修改为 363K，

如图 18-4 所示。

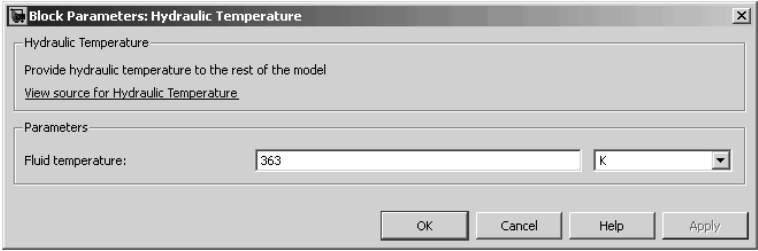


图 18-4 修改液体温度

(13) 再次仿真可以看到示波器温度变为 363K。

18.3 本例小结

本例以机载简易温度测量系统为对象介绍如何基于 Simscape 语言创建新的域模型和部件模型，通过本例学习应对 Simscape 域模型和部件模型结构的掌握进一步加深，并熟悉 Simscape 文件的路径系统。



第 19 例 船舶温度调节系统



前面两例学习了如何基于 Simscape 语言创建多领域模型并进行仿真的内容，事实上多领域模型中除基于 Simscape 语言的模块外，还存在基于 Simulink 的模块，这两类模块相互配合，能够有效地提高系统建模与仿真的速度。前面已经学过了基于 M 语言的 S 函数建立方法，鉴于 S 函数的重要性，以及考虑到前后版本 S 函数的变化，本例和下一例继续学习 Level-2 S 函数与基于 C 语言的 S 函数。通过本例学习，应该掌握以下两点：

- Level-2 S 函数使用方法。
- 基于 Level-2 S 函数船舶温度调节系统设计。

19.1 Level-2 S 函数简介

19.1.1 Level-2 S 函数基本特性

Level-2 的 MATLAB S 函数允许用户使用 MATLAB 语言来创建自定义块具有多个输入和输出端口，能够处理任何类型的 Simulink 模型，包括任何数据类型的矩阵和帧信号的信号。相比于第一代的 S 函数，Level-2 函数能够实现更多的功能，比如在多输入多输出等方面。

MATLAB 函数本身由一组回调方法组成，在 Simulink 引擎调用时更新或模拟模型。在实际工作中，回调方法执行初始化和计算块的输出 S 函数的定义。

为了方便这些任务时，引擎将运行时对象作为参数的回调方法。运行时有效地作为一个 MATLAB 的 S 功能块的代理对象，允许在模拟或模型更新的回调方法来设置和访问块属性。

当仿真引擎调用一个 Level-2 MATLAB 的 S 函数的回调方法，它通过调用 Simulink.MSFcnRunTimeBlock 类的一个实例的作为参数。Simulink.MSFcnRunTimeBlock 类被称为 S 功能块对象的运行时间。这种情况下，Level-2 的 MATLAB 的 S 函数回调方法的 SimStruct 结构与用于 C MEX S 函数的回调方法有异曲同工之妙。对通过端口获取参数、状态和工作载体的各种元素的提供和获取信息方法的读者请参阅 Simulink.MSFcnRunTimeBlock 类信息获取和设置运行时对象的属性和调用运行时对象方法的文档。



Level-2 S 函数运行时不支持 MATLAB 稀疏矩阵对象。例如，如果变量块是如下行一个运行时对象，Level-2 MATLAB 的 S 函数将产生一个错误：

```
block.Outport(1).Data = speye(10);
```

其中的 `speye` 命令形成矩阵稀疏的身份。

注意：其他的 MATLAB 程序，除了 MATLAB 的 S 函数可以使用运行时对象来获取信息关于 MATLAB 的 S 函数的模型，该模型模拟。在模拟中使用 Simulink 的更多信息，请参阅访问块数据。

19.1.2 Level-2 S 函数模板

使用 Level-2 MATLAB 的基本的 S 函数模板 `msfuntmpl_basic.m` 的头开始创建一个新的 Level-2 MATLAB 的 S 函数。该模板包含通过 Level-2 MATLAB 的 S-API 函数定义所需的回调方法的实现框架。要编写一个更复杂的 S 函数，使用注释的模板 `msfuntmpl.m`。

要创建一个 MATLAB 的 S 函数，做一个复制的模板和编辑的副本，以反映所建立的 S 函数的期望行为。下面两节描述 MATLAB 代码模板的内容。一节编写一个 Level-2 MATLAB S 函数的示例，另一节介绍了如何编写一个 Level-2 MATLAB 的 S 函数模型的单位延迟。

1. Level-2 S 函数回调方法

Level-2 的 MATLAB S-API 函数定义构成一个 Level-2 MATLAB 的 S 函数的回调方法的签名和一般用途。S 函数本身提供这些回调方法的实现，进而确定块的属性（例如，端口、参数和状态）和行为（例如，块输出作为时间的函数和功能块的输入、状态和参数）。通过创建一个 S 函数与一组适当的回调方法，用户可以定义一个块类型，以满足应用程序的具体要求。

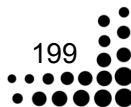
一个 Level-2 MATLAB S 函数必须至少包括以下的回调方法：

- 初始化设置功能设定为 S 函数的基本特性。
- 一个输出功能，计算的 S 函数输出。

除以上两个外，S 函数还可以包含其他方法，用户可根据该块的功能以及设计要求，在 S 函数中进行定义。

下面列出了所有的 Level-2 MATLAB 的 S 函数回调方法。

```
setup mdlInitializeSizes  
CheckParametersmdlCheckParameters  
DerivativesmdlDerivatives  
DisablemdlDisable  
EnablemdlEnable  
InitializeConditionmdlInitializeConditions  
OutputsmdlOutputs
```





```
PostPropagationSetupmdlSetWorkWidths
ProcessParametersmdlProcessParameters
ProjectionmdlProjection
SetInputPortComplexSignalmdlSetInputPortComplexSignal
SetInputPortDataTypemdlSetInputPortDataType
SetInputPortDimensionsmdlSetInputPortDimensionInfo
SetInputPortDimensionsModeFcnmdlSetInputPortDimensionsModeFcn
SetInputPortSampleTimemdlSetInputPortSampleTime
SetInputPortSamplingModemdlSetInputPortFrameData
SetOutputPortComplexSignalmdlSetOutputPortComplexSignal
SetOutputPortDataTypemdlSetOutputPortDataType
SetOutputPortDimensionsmdlSetOutputPortDimensionInfo
SetOutputPortSampleTimemdlSetOutputPortSampleTime
SimStatusChangemdlSimStatusChange
StartmdlStart
TerminatemdlTerminate
UpdatemdlUpdate
WriteRTWmdlRTW
```

选择好合适的回调方法后需要正确的使用,Level-2 MATLAB 的 S 函数初始化相应的 Level-2 的 MATLAB 的 S 功能块的实例。在这方面,设置方法是类似 C MEX S 函数的 mdlInitializeSizes 和 mdlInitializeSampleTimes 实现的回调方法。设置方法执行以下任务:

- 初始化该块的输入和输出端口的数量。
- 设置属性,如尺寸,数据类型,复杂性,这些端口的采样时间。
- 指定块的采样时间。请参阅 Simulink 文档如何指定有效的采样时间。
- 设置 S 函数对话框中的参数数量。
- 注册 S 回调函数。

2. 编写 S 函数

下面的步骤说明了如何编写一个简单的 Level-2 的 MATLAB 的 S 函数。使用时,相应步骤可以参考模型 msfcdemo_sfundsc2.mdl 中 msfcn_unit_delay.m 的使用例子。

复制 Level-2 MATLAB 的 S 函数模板 msfuntmpl_basic.m 到自己的工作文件夹中。

注意:当复制文件时,如果更改文件名,请同时更改文件中函数名为相同的名称。

通过修改设置的方法来初始化 S 函数的属性。对于这个例子:

- 设置运行时对象的 NumInputPorts 和 NumOutputPorts 的属性,以 1 初始化一个输入端口和一个输出端口。
- 调用运行时对象的 SetPreCompInPortInfoToDynamic 方法和 SetPreCompOutPortInfoToDynamic 方法,输入和输出端口从模型继承其编译属性(尺寸、数据类型、复杂性和采样模式)。
- 设置运行时对象的 InputPort 的 DirectFeedthrough 属性为 false,以指示输入端口没有直接馈通。



- 将 S 函数初始化对话框运行时对象 NumDialogPrms 的属性设置为 1。
- 将 S 函数继承的采样时间设置运行时对象 SampleTimes 设为 0。
- 调用运行时对象 RegBlockMethod，来注册以下四个 S 函数中使用的回调方法。

```
PostPropagationSetup
InitializeConditions
    Output
    Update
```

- 从模板文件的副本中删除其他回调方法。在 RegBlockMethod 调用中，第一个输入参数是 S 函数的 API 方法的名称，第二个输入参数的功能是处理在 MATLAB 中与 S 函数相关联的本地函数。

下面给出基于 msfcn_unit_delay.m 修改得到初始化的例子：

```
%%注册一个单独的对话框参数
block.NumDialogPrms = 1;

%%寄存器数量的输入和输出端口
block.NumInputPorts = 1;
block.NumOutputPorts = 1;

%%设定功能端口属性动态继承
%%。
block.SetPreCompInPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

%%硬编码的某些端口属性
block.InputPort 尺寸(1) = 1;
block.InputPort(1).DirectFeedthrough = FALSE;

block.OutputPort 尺寸(1) = 1;

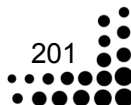
%%设定块样品的时间来继承
block.SampleTimes = [-1 0];

%%的注册方法
block.RegBlockMethod('PostPropagationSetup', @DoPostPropSetup);
block.RegBlockMethod('InitializeConditions', '@InitConditions');
block.RegBlockMethod('输出', @输出);
block.RegBlockMethod('更新', @更新);
```

如果 S 函数需要连续状态，初始化连续状态中的安装方法，使用的运行时对象的 NumContStates 的数值，不要初始化离散状态。

初始化的离散状态在 PostPropagationSetup 中进行修改。一个 Level-2 MATLAB 的 S 函数离散状态信息在 DWork 载体中。

采用 PostPropagationSetup 方法，名为 DoPostPropSetup，从 msfcn_unit_delay.m 初始





化 DWork，载体的名称为 X0。

以下是功能 DoPostPropSetup 的程序实例：

```
%设置 Dwork
block.NumDworks= 1;
block.Dwork(1)。名称= x0 来进行;
block.Dwork 尺寸(1)= 1;
block.Dwork(1)。DatatypeID= 0;
block.Dwork(1)。='真正'的复杂性;
block.Dwork(1)。UsedAsDiscState= TRUE;
```

如果 S 函数使用额外的 DWork 向量，在 PostPropagationSetup 中进行初始化。

初始化值的离散和连续状态或其他 DWork 的载体在 InitializeConditions 或开始回调方法。在开始的模拟值初始化一次使用开始回调方法。

注意：使用 InitializeConditions 时，每当含有的 S 函数启用的子系统被重新启用，需要重新初始化相应的值。

在这个例子中，使用 InitializeConditions 方法来设置 S 函数的对话框参数值的离散状态的初始条件。例如，InitializeConditions 在 msfcn_unit_delay.m 方法是：

```
%初始化 Dwork
block.Dwork block.DialogPrm.number;
```

S 函数与连续状态，使用 ContStates 的运行时对象的方法来初始化连续状态日期，例如：

```
block.ContStates.Data = 1.0;
```

在输出回调方法中计算 S 函数的输出。在这个例子中，设置输出离散状态的当前值存储在 DWork 矢量。

在 msfcn_unit_delay.m 的输出方法是：

```
block.OutputPort.number= block.Dwork.number;
```

对于连续状态的 S 函数，计算出的状态离散工具的回调方法。离散数据存储在运行时对象及其离散属性中。例如，下面的行设置的第一输入信号的值等于在第一状态离散值。

```
block.Derivatives.number = block.InputPort.number;
```

这个例子不使用连续的状态，因此，没有实现连续工具的回调方法。

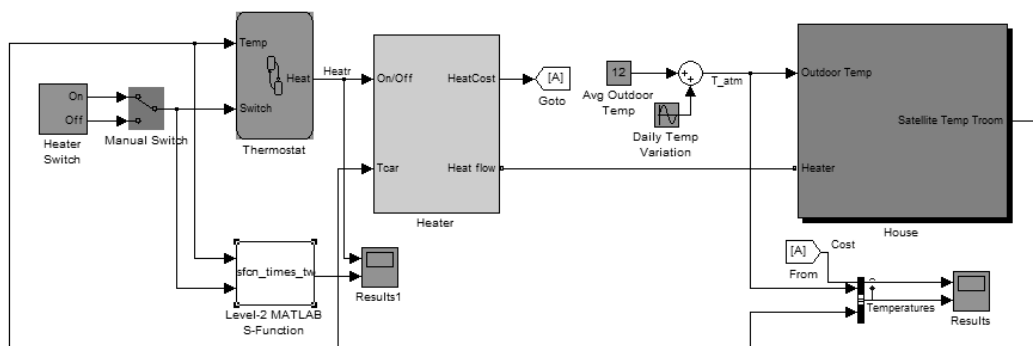
19.2 船舶温度调节系统

在第 17 例基础上进行修改得到船舶温度调节系统。该系统结构与第 17 例结构上相似，主要的不同在于：本例仿真中添加了一个 Level-2 S 函数，该函数的输入与 Stateflow 模块相同，输出也可以作为温度条件控制输入。仿真结果将基于 Stateflow 生成的温度控



制指令与基于 S 函数生成的控制指令进行对比从而判断基于 S 函数生成的模块是否满足功能需求。

建立的船舶温度调节系统如图 19-1 所示。用于建立 Level-2 S 函数的代码在图 19-1 下方。仿真结果如图 19-2 所示。由图可以看出基于 S 函数给出的温度调节指令离散特性做得不好，需要进一步修改，感兴趣的读者可以自己尝试一下。



Sail Heating System

The demo represents a simple sail heating system consisting of a heater, thermostat, and a sail structure with four thermally distinguishable parts: inside air, house walls, windows, and roof. The sail exchanges heat with the environment through its walls, windows, and roof. Each path is simulated as a combination of a thermal convection, thermal conduction, and the thermal mass. The heater starts pumping hot air if room temperature falls below 18 deg C and is turned off if temperature exceeds 23 deg C. As a result, you can monitor the temperatures of house parts and heat losses through them. The manual switch allows you to investigate system behavior with the heating system turned off.

图 19-1 船舶温度调节系统

Level-2 S 函数代码如下：

```
function msfcn_times_two(block)
% Level-2 MATLAB file S-Function for times two demo.
% Copyright 1990-2009 The MathWorks, Inc.
% $Revision: 1.1.6.2 $
    setup(block);

%endfunction

function setup(block)

%% Register number of input and output ports
block.NumInputPorts = 1;
block.NumOutputPorts = 1;

%% Setup functional port properties to dynamically
%% inherited.
block.SetPreCompInpPortInfoToDynamic;
block.SetPreCompOutPortInfoToDynamic;

block.InputPort(1).DirectFeedthrough = true;
```





```
%% Set block sample time to inherited
block.SampleTimes = [-1 0];

%% Set the block simStateCompliance to default (i.e., same as a built-in
block)
block.SimStateCompliance = 'DefaultSimState';

%% Run accelerator on TLC
block.SetAccelRunOnTLC(true);

%% Register methods
block.RegBlockMethod('Outputs', @Output);

%endfunction

function Output(block)

    block.OutputPort(1).Data = 2*block.InputPort(1).Data;

%endfunction
```

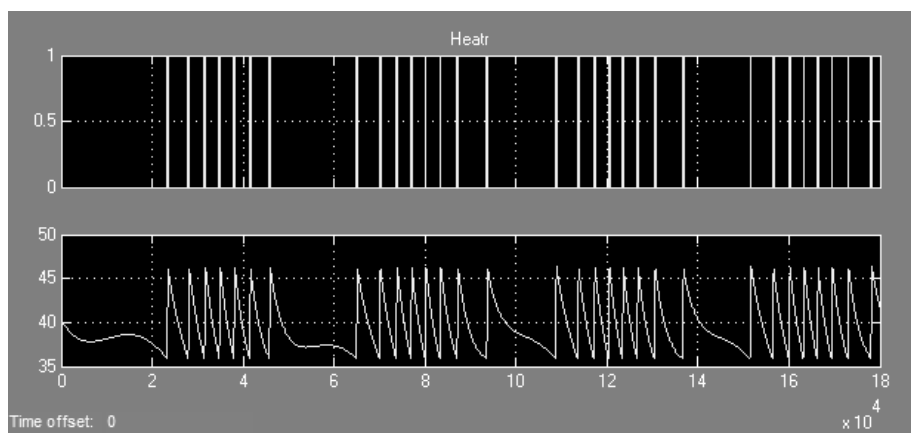


图 19-2 船舶温度调节系统仿真结果

19.3 本例小结

本例介绍了 Level-2 S 函数结构及建立过程。Level-2 S 函数相对于第一代 S 函数能够实现更多的功能,比如多输入多输出等,在使用方面具有更大的便利性,而且后续 Simulink 版本将逐步不支持第一代 S 函数,因此有必要了解 Level-2 S 函数。



第 20 例 卫星温度调节系统

本例以卫星为对象，通过基于 C 语言的 S 函数与 Simscape 多领域模块建立卫星温度调节系统仿真模型，并进行仿真。通过本例学习，应掌握以下两点：

- 基于 C 语言 S 函数结构。
- 基于 S 函数生成器生成 C 语言 S 函数过程。

20.1 基于 C 语言的 S 函数简介

基于 C 语言的 S 函数相比于基于 M 语言的 S 函数具有更高的效率和更快的速度，在涉及多领域的大型仿真中，仿真时间是一个评价仿真系统好坏的重要指标。传统上由于 MATLAB 基于解释性编译方式，因此在仿真时间上相比于其他编程语言建立的仿真系统要长，为既能利用到 MATLAB 丰富的库函数，又能减少仿真运行时间，可以基于 C 语言编写 S 函数，这将使得仿真系统具有更高的效率。

一个典型的基于 C 语言的 S 函数结构如下所示：

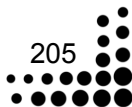
```
The general format of a C MEX S-function is shown below:
#define S_FUNCTION_NAME  your_sfunction_name_here
#define S_FUNCTION_LEVEL 2
#include "simstruc.h"

static void mdlInitializeSizes(SimStruct *S)
{
}

<additional S-function routines/code>

static void mdlTerminate(SimStruct *S)
{
}

#ifdef MATLAB_MEX_FILE    /* Is this file being compiled as a
                           MEX-file? */
#include "simulink.c"      /* MEX-file interface mechanism */
#else
#include "cg_sfun.h"       /* Code generation registration
                           function */
#endif
#endif
```





mdlInitializeSizes 是 Simulink 调用 S 函数时的第一个例程。随后的 Simulink 继续调用 S 函数其他方法（所有 MDL）。在一个仿真周期结束后，调用 mdlTerminate。

用户可以通过以下三种方法创建 C 语言 S 函数：

- 手写 S 函数——可以手动输入 C 语言 S 函数全部代码。此时可以参考 MATLAB 提供的 C 语言例子中的结构与内容，然后可以以例子作为一个起点，创建 C 语言 S 函数。
- S 函数生成器——该种方式下，系统从一个事先制作好的半定制图形用户界面来建立一个 C 语言 S 函数。这种方式下用户无需在写 S 函数时从头开始。S 函数生成器的更多信息将在下面详细介绍。
- 重用代码工具——此该种方式是通过生成库的方式来调用。

由于 S 函数生成器生成器具有简单方便和半定制等优点，使用其可以迅速生成用户想要的 C 语言 S 函数，因此这里对这种方法予以详细介绍。

S 函数生成器其实是 Simulink 中的一个模块，通过单击 Simulink 工具栏中“Simulink|User Defined Functions”，在右侧将会出现该模块。

将该模块拖拽到新建模型中，如图 20-1 所示。

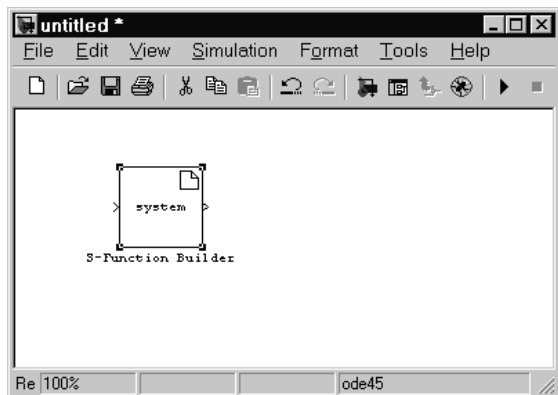


图 20-1 S 函数生成器模块

下面对基于 S 函数生成器模块进行 S 函数设计进行介绍。

1. 初始化设计

在 S 函数生成器中，用户可以指定某些属性从而通过 S 函数生成器自动生成 S 函数。双击 S 函数生成器模块图标，或选中模块，并单击右键从上下文菜单中的“编辑”菜单上选择打开块，此时将出现一个对话框，如图 20-2 所示。图中显示的是 S 函数初始化属性设置界面。

从图中可以看出，初始化设置界面可以对 S 函数的名称、离散状态数量、连续状态数量、采样模式等进行设置。

2. 数据属性设置

设置完初始化界面属性后，通过单击“Data Properties”可以打开数据属性界面，

如图 20-3 所示。

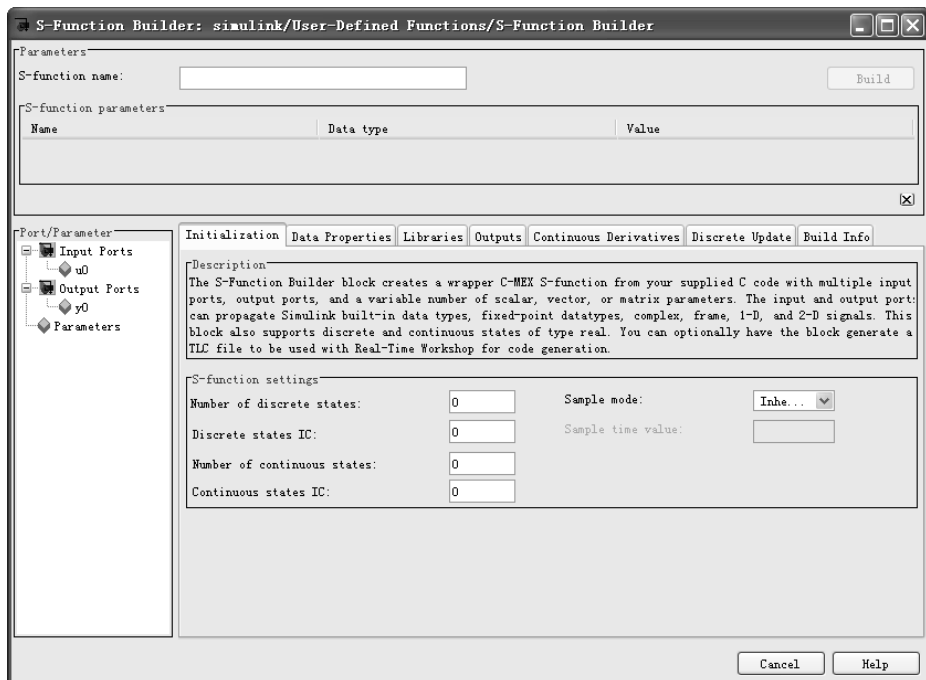


图 20-2 S 函数初始化属性设置界面

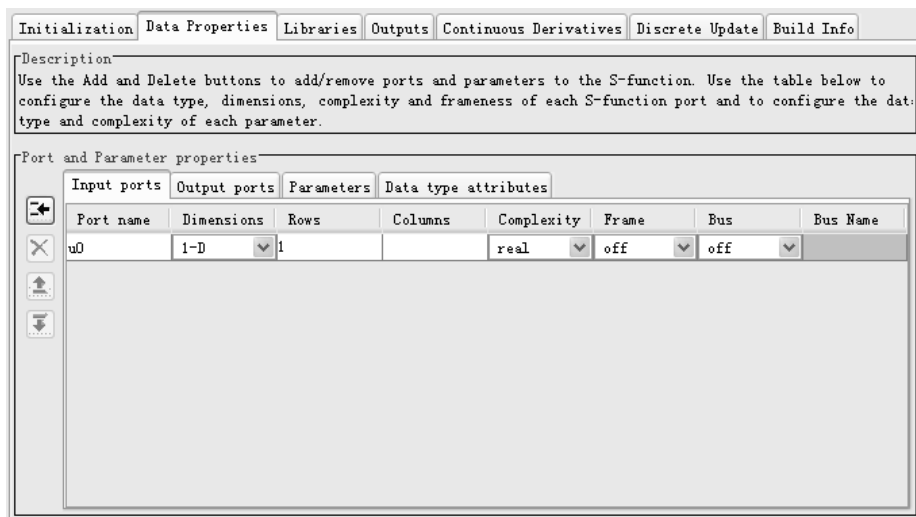


图 20-3 S 函数输入端口设置界面

从该界面可以看出，数据属性界面分为四个子界面：输入端口界面、输出端口界面、参数端口界面和数据类型属性界面。下面对这四个子界面分别予以介绍。

(1) 输入端口界面，其内容见图 20-3。

列的左侧窗格的按钮允许用户添加、删除或重新排序端口或参数，根据当前选择的



窗格。要添加一个端口或一个参数，单击“添加”按钮（最上端的按钮列中的按钮）。要删除当前选中的端口或参数，单击“删除”按钮（位于“添加”按钮下方）。若要将当前选定的端口或参数，在相应的 S 功能端口或参数列表中向上移动一个位置，则单击向上按钮（“删除”按钮的下方）。若要将当前选定的端口或参数，在相应的 S 功能端口或参数列表中向下移动一个位置，单击向下键（向上按钮的下方）。

该窗格还包含标签化的窗格，使用户可以指定所创建的端口和参数的属性。端口可分为以下几类：

- 输入端口界面。
- 输出端口界面。
- 参数界面。
- 数据类型属性界面。
- 输入端口界面。

其中输入端口界面可让用户检查并修改其属性的 S 函数的输入端口。

界面中包括一个可编辑的表格，列出的顺序输入端口，在该端口上出现的 S 函数生成器块的属性。表中的每一行对应一个端口。该行中的每个条目显示端口的一个属性。

- 端口名称。编辑此字段更改端口名称。
- 维数。该端口用于设置所接受的输入信号的维数。要支持的维数通过显示一个列表来让用户选择。通过点击旁边的按钮，从下拉列表选择一个新值，来指定动态信号的大小。
- 行。指定输入信号行的数目。
- 列。指定输入信号列的数目。

注意：对于输入信号的两个维度，如果行维度的大小是动态的，则列维度也必须是动态尺寸或设置为 1。如果列的尺寸设定为其他值，因为这是一个无效的尺寸规格，虽然 S 函数进行编译，但 S 函数将不进行任何模拟运行。

- 复杂性。指定的输入端口是否接受实数或复数值的信号。
- 框架。指定该端口是否接受基于帧的信号所产生的信号处理模块或通信模块产品。
- 总线。如果输入信号的 S 函数生成器模块是总线或总线列，则使用下拉菜单中选择“打开”。
- 总线名称。S 函数会自动为用户创建一个总线对象，如果用户的输入信号是总线。在提供的字段中的总线名称列中，输入用户定义的总线名称。

（2）输出端口界面。

输出端口界面可让用户检查并修改其属性的 S 函数的输出端口。

注意：对于输出信号，具有两个维度，如果其中一个维度的大小是动态的，则其他尺寸也必须是动态尺寸或设置为 1。如果设置为其他值，S 函数虽然会编译，但不会产生任何模拟运行。

（3）参数界面。参数窗格可让用户检查并修改其属性的 S 函数的参数。

（4）数据类型属性界面，如图 20-4 所示。

此界面允许用户指定输入和输出端口数据类型的属性。

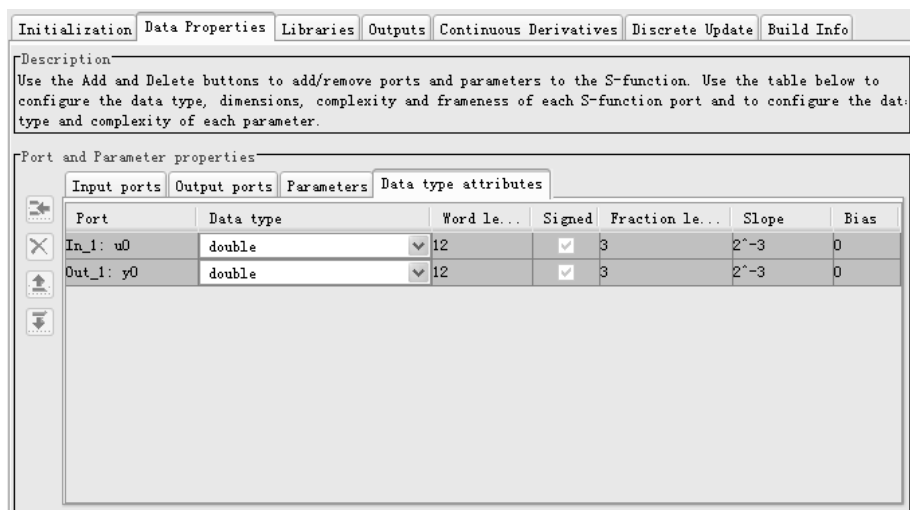


图 20-4 S 函数数据类型设置界面

从图中可以看出。界面中包含一个表，表中列出 S 函数端口的数据类型属性。用户可以编辑表中的某些字段，其他字段是灰色的。每一行对应于目标 S 函数的一个端口，每一列指定一个属性对应的端口。下面对端口属性进行介绍：

- 端口名称。此字段显示的输入端口和输出端口窗格中输入的名称。用户可以编辑此字段。
- 数据类型。单击旁边的按钮，以显示支持的数据类型的列表。要更改数据类型，从列表选择一个不同的数据类型。

3. 静态库设置

静态库界面如图 20-5 所示。静态库界面可让用户设置所引用的外部代码文件以及指定相应的位置。

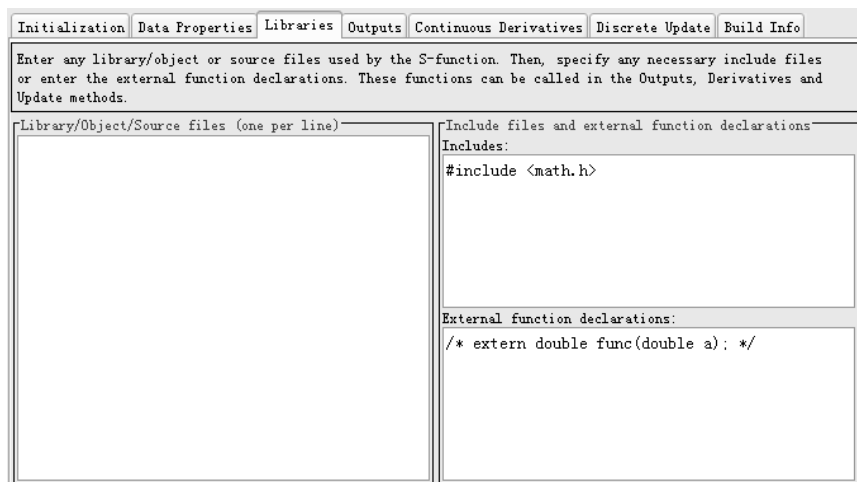
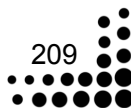


图 20-5 S 函数静态库设置界面





4. 输出设置

输出设置界面如图 20-6 所示。在计算每个仿真时间步长中 S 函数输出时，即使用输出界面中输入的代码。

Initialization Data Properties Libraries **Outputs** Continuous Derivatives Discrete Update Build Info

Code description
Enter your C-code or call your algorithm. If available, discrete and continuous states should be referenced as, `xD[0]...xD[n]`, `xC[0]...xC[n]` respectively. Input ports, output ports and parameters should be referenced using symbols specified in the Data Properties. These references appear directly in the generated S-function.

```
/* This sample sets the output equal to the input
   y0[0] = u0[0];
For complex signals use: y0[0].re = u0[0].re;
                        y0[0].im = u0[0].im;
                        y1[0].re = u1[0].re;
                        y1[0].im = u1[0].im;*/
```

☒ Inputs are needed in the output function(direct feedthrough)

图 20-6 S 函数输出设置界面

5. 连续状态微分设置

连续状态微分设置界面如图 20-7 所示。

Initialization Data Properties Libraries Outputs **Continuous Derivatives** Discrete Update Build Info

Code description
This section is optional and use to calculate the derivatives. It is called only if the S-function has one or more states. The states and derivatives of the S-function are of type double and must be referenced as `xC[0]`, `xC[1]`, and `dx[i]` etc respectively. Input ports, output ports and parameters should be referenced using the symbols specified in the Data Properties. These references appear directly in the generated S-function.

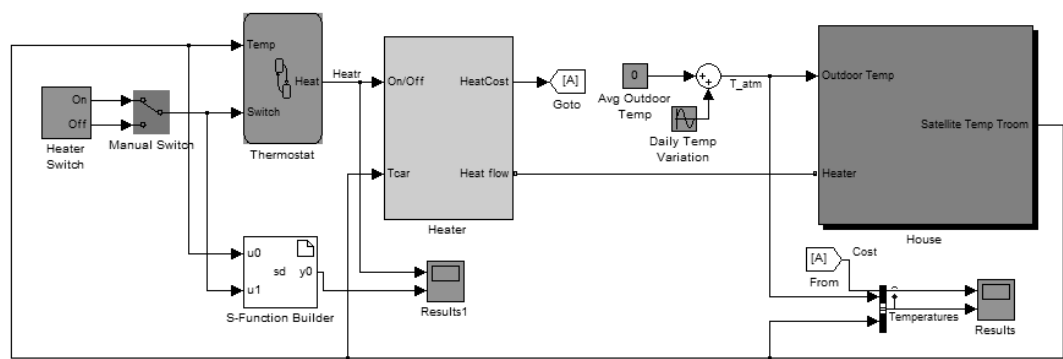
```
/*
 * Code example
 *   dx[0] = xC[0];
*/
```

图 20-7 S 函数连续状态微分设置界面

如果 S 函数有连续状态，则在连续状态微分设置界面中需要输入连续状态的微分表达式。

20.2 卫星温度调节系统建模与仿真

同样是在第 17 例基础上进行修改得到卫星三维模型，如图 20-8 所示（源文件请见提供下载的配书资源）。相比于第 17 例，本例增加了一个 S 函数生成器模块，该函数的输入与 Stateflow 模块相同，输出也可以作为温度条件控制输入。仿真结果将基于 Stateflow 生成的温度控制指令与基于 C 语言的 S 函数生成的控制指令进行对比从而判断基于 C 语言的 S 函数生成的模块是否满足功能需求。



Satellite Heating System

The demo represents a simple satellite heating system consisting of a heater, thermostat, and a satellite structure with four thermally distinguishable parts: inside air, house walls, windows, and roof. The satellite exchanges heat with the environment through its walls, windows, and roof. Each path is simulated as a combination of a thermal convection, thermal conduction, and the thermal mass. The heater starts pumping hot air if room temperature falls below 18 deg C and is turned off if temperature exceeds 23 deg C. As a result, you can monitor the temperatures of house parts and heat losses through them.
The manual switch allows you to investigate system behavior with the heating system turned off.

图 20-8 卫星温度调节系统

仿真结果如图 20-9 所示，由仿真图可以看出，基于 C 语言的 S 函数离散特性也较好，将其应用到温度调节中也能完成任务。

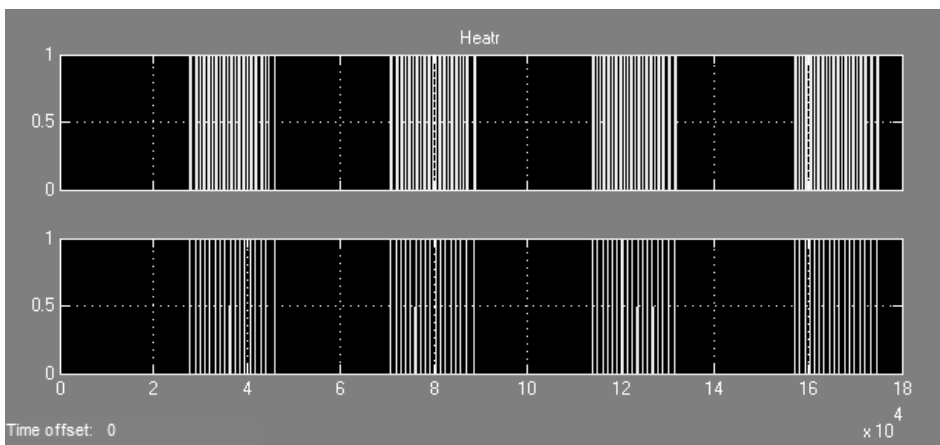


图 20-9 卫星温度调节系统仿真结果

基于 S 函数生成器得到的 C 文件代码如下：



```
#if defined(MATLAB_MEX_FILE)
#include "tmwtypes.h"
#include "simstruc_types.h"
#else
#include "rtwtypes.h"
#endif

/* %%-SFUNWIZ_wrapper_includes_Changes_BEGIN --- EDIT HERE TO _END */
#include <math.h>
/* %%-SFUNWIZ_wrapper_includes_Changes_END --- EDIT HERE TO _BEGIN */
#define u_width 1
#define y_width 1
/*
 * Create external references here.
 */
/* %%-SFUNWIZ_wrapper_externs_Changes_BEGIN --- EDIT HERE TO _END */
/* extern double func(double a); */
/* %%-SFUNWIZ_wrapper_externs_Changes_END --- EDIT HERE TO _BEGIN */

/*
 * Output functions
 */
void sd_Outputs_wrapper(const real_T *u0,
                        const real_T *u1,
                        real_T *y0,
                        const real_T *xC)
{
/* %%-SFUNWIZ_wrapper_Outputs_Changes_BEGIN --- EDIT HERE TO _END */
/* This sample sets the output equal to the input
   y0[0] = u0[0];
For complex signals use: y0[0].re = u0[0].re;
                        y0[0].im = u0[0].im;
                        y1[0].re = u1[0].re;
                        y1[0].im = u1[0].im;*/
if ((u0[0]<18)&&(u1[0])) y0[0]=1;
else y0[0]=0;

/* %%-SFUNWIZ_wrapper_Outputs_Changes_END --- EDIT HERE TO _BEGIN */
}

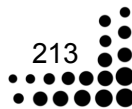
/*
 * Derivatives function
 */
void sd_Derivatives_wrapper(const real_T *u0,
                            const real_T *u1,
```



```
        const real_T *y0,  
        real_T *dx,  
        real_T *xC)  
{  
    /* %%%-SFUNWIZ_wrapper_Derivatives_Changes_BEGIN --- EDIT HERE TO _END  
*/  
  
    /* %%%-SFUNWIZ_wrapper_Derivatives_Changes_END --- EDIT HERE TO _BEGIN  
*/  
}
```

20.3 本例小结

本例介绍了 C 语言 S 函数的结构，并介绍基于 S 函数生成器建立 C 语言 S 函数的过程。通过仿真对比可以看出 C 语言 S 函数可以实现与 Stateflow 类似的功能。实际上，Simulink 是基于 C 语言编写，相比于基于 M 语言生成的 S 函数，C 语言 S 函数不但效率高，还可以实现更多的功能。



第六部分 图像工程仿真实例



引言——图像处理工具箱介绍（上）

随着数字化时代的到来，图像处理在实际工程中得到越来越广泛的使用，图像处理在计算机、电子、通信、气象和医学等众多领域中起着越来越重要的作用。MATLAB 的图像处理工具箱提供了丰富的图像处理功能，而且基于 MATLAB 还可以自定义算法，从而为工程当中的快速应用奠定了基础。

利用图像处理工具箱，可以完成以下几部分功能。

- (1) 图像合成。可以实现图像的代数运算和逻辑运算。
- (2) 空间变换。可以对图像进行旋转、缩放和裁剪等操作。
- (3) 邻域和块处理。可以进行块处理操作、滑动邻域操作、分离块操作和列处理。
- (4) 线性滤波和滤波器设计。可以设计线性滤波器和 FIR 等滤波器。
- (5) 基于区域进行处理。可以指定区域并对区域进行滤波和填充。
- (6) 变换域处理。可以进行傅里叶变换、离散余弦变换和 Radon 变换。
- (7) 数学形态学运算。可以进行膨胀和腐蚀以及数学形态重建等操作。
- (8) 图像分析。可以进行灰度统计、边缘检测、边界跟踪和四叉树分解等操作。
- (9) 图像恢复。可以利用各种滤波器和算法恢复图像。

图像处理工具箱支持索引图像、灰度图像、二值图像和 RGB 图像等四种基本的图像类型。

1. 索引图像

索引图像由数据矩阵 X 和映射矩阵 map 组成，数据矩阵可以是 `uint8`、`uint16` 或 `double` 型的。映射矩阵是一个 `double` 型的 $m \times 3$ 数组，元素为 0 至 1 之间的浮点值。 map 矩阵的每一行指定某种颜色的红色、绿色和蓝色组分。索引图像将像素值与 map 矩阵的值直接进行映射。每个图像元素的颜色是通过将 X 的对应值作为索引编号，从 map 矩阵中得到的。值 1 指向 map 矩阵的第一行，值 2 指向第二行，以此类推。

映射矩阵通常与索引图像一起保存，并且在用 `imread` 函数载入图像时自动载入。图像矩阵和颜色矩阵查找表之间的关系取决于图像矩阵的类型。如果图像矩阵是 `double` 的，



值 1 指向颜色查找表的第一行,值 2 指向第二行,以此类推。如果图像矩阵是 unit8 或 unit16 型的,就会存在偏差,此时值 0 指向颜色查找表的第一行,值 1 指向第二行。偏移还用在图形文件格式中,使得可以支持的颜色个数最大化。

注意:对于 unit16 类型,工具箱只提供了有限的支持。可以将这种类型的数据读入 MATLAB 并且显示它们,但在处理 unit16 类型的索引图像之前,必须先将它转换为 double 型或 unit8 型。

2. 灰度图像

灰度图像由一个数据矩阵 I 来表示,矩阵中的值表示一定范围内的亮度值。MATLAB 将一幅灰度图像保存为一个单一的矩阵,矩阵的每个元素对应于一个图像像素。矩阵可以是 double、unit8 或 unit16 的。

亮度矩阵中的元素表示不同的亮度或灰度级,其中亮度 0 表示黑色,亮度 1255 或 65535 表示饱和亮度或白色。

3. 二值图像

在二值图像中,假设每个像素取两个离散值中的一个,分别对应 on 和 off,二值图像保存为 logical 数组,值为 0 或者 1。

4. RGB 图像

RGB 图像有时称为真彩色图像,在 MATLAB 中保存为 $m \times n \times 3$ 的数据数组,定义每个单独像素的红色、绿色和蓝色组分。RGB 图像不使用调色板。每个像素的颜色由像素位置上红色、绿色和蓝色亮度的组合确定。RGB 图像是 24 位图像,其中红色、绿色和蓝色组分均为 8 位。这将产生一千六百多万种颜色。采用这些颜色,在精度上可以逼近现实场景中图像的真实颜色。所以 RGB 图像又称为真彩色图像。

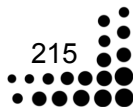
RGB 数组可以是 double、unit8 或 unit16 型的。在 double 型的 RGB 数组中,每一个颜色组分的值取 0 和 1 之间的数。一个颜色组分为 (0,0,0) 的像素显示为黑色,颜色组分为 (1,1,1) 的像素显示为白色。每个像素的这 3 种颜色组分保存在数据数组的第三维上。例如,像素 (10,5) 的红色、绿色和蓝色组分分别保存在 RGB(10,5,1),RGB(10,5,2) 和 RGB(10,5,3)。

下面分别对图像类型转换、图像数据和如何读取图像数据进行介绍。

一、图像类型转换

对于有些操作,需要将图像转换为不同的图像类型。例如,如果试图对一幅保存为索引格式的彩色图像进行滤波,应该先将其转换为 RGB 格式,对 RGB 图像使用滤波器时,MATLAB 将对图像中的亮度值进行滤波,直到合适为止。如果试图对索引图像滤波,结果可能没有意义。

注意:如果将图像从一种格式转换为另一种格式,产生的图像看起来可能会与原来





的图像不同。例如，把一幅彩色索引图像转换为灰度图像，产生的图像可能是彩色图像，而不是灰度图像。

图像类型转换函数如表 F1 所示。

表 F1 图像类型转换函数

函 数	描 述
dither	采用抖动的方法，把灰度图像转换为二值图像，或者把 RGB 图像转换为索引图像
gray2ind	把灰度图像转换为索引图像
grayslice	利用给定的灰度阈值，将灰度图像转换为索引图像
im2bw	利用给定的灰度阈值，将灰度图像、索引图像或 RGB 图像转换为二值图像
ind2rgb	将索引图像转换为灰度图像
mat2gray	通过比例化数据，将矩阵中的数据转换为灰度图像
rgb2gray	将 RGB 图像转换为灰度图像
rgb2ind	将 RGB 图像转换为索引图像

还可以用 MATLAB 语句进行一定的图像转换操作，例如，可以通过沿第三维聚合原始矩阵的 3 个复制把一幅灰度图像转换为 RGB 格式：

$$\text{RGB} = \text{cat}(3, \text{I}, \text{I}, \text{I});$$

红色、绿色和蓝色面板具有相同的矩阵，所以该 RGB 图像显示为灰色。

二、图像数据

该部分包含图像数据保存类型和图像数据读写等方面的内容。

MATLAB 中基本的数据结构是数组，它表示一个经过排序的实数或虚数元素集合。它适用于图像数据的表示，因为图像数据是一系列经过排序的颜色或灰度数据的集合。

MATLAB 将大部分图像保存为二维数组，其中，数组的一个元素对应于所显示的图像中的一个像素。例如，一个由 200 行 300 列不同颜色的点组成的图像，在 MATLAB 中保存为一个 200×300 的矩阵。有些图像，比如 RGB 图像，需要用一个三维数组进行保存，其中第三维的第 1 个面表示红色的像素亮度，第 2 个面表示绿色的像素亮度，第 3 个面表示蓝色的像素亮度。这一特点使得在 MATLAB 在处理图像与处理其他矩阵数据类型类似，并且使得 MATLAB 的所有函数都可以用于图像处理。例如，可以用一般的矩阵角标索引从图像矩阵中选择一个单独的像素。正常情况下，MATLAB 把大部分数据保存为 double 型数组，这些数组中的数据保存为双精度浮点型，所有 MATLAB 函数都可以处理这些数组。

为减小内存需求，MATLAB 支持将图像数据保存为 8 位或 16 位无符号整型数组，这些数组只需要 double 型数组八分之一或四分之一的内存。

对于 unit8 和 unit16 两种类型的图像数据，可以进行许多标准的 MATLAB 数组操作，包括：

- 索引，包括逻辑索引。
- 重塑、排序和聚合。
- 从 Mat 文件读取数据或者把数据写入 Mat 文件。
- 使用关系操作符。

有些 MATLAB 函数，接受 unit8 或 unit16 类型的数据，但是返回双精度数据。

基本的算术操作符不接受 unit8 或 unit16 类型的数据，因为这些算术操作是许多图像处理操作的一个重要部分，所以图像处理工具箱中包含了支持对 unit8 和 unit16 及其他类型数据进行这些操作的函数。

表 F2 所列 MATLAB 根据不同的图像类型和存储类型将数据矩阵元素解释为像素颜色的方式。

表 F2 矩阵元素解释为像素颜色的方式

图 像 类 型	保 存 类 型	解 释
二值图像	logical	元素值为 0 和 1 的数组
索引图像	double	元素值为[l,p]中整数的数组
	unit8,unit16	元素值为[0,p-1]中整数的数组
灰度图像	double	元素值为浮点值的数组，值一般在[0,1]中取值
	unit8,unit16	整型数组，值一般在[0,255]或[0,65535]中取值
RGB 图像	double	m×n×3 的数组，元素值为[0,1]中浮点数
	unit8,unit16	m×n×3 的数组，元素为[0,255]或[0,65535]中的整型值

三、读/写图像数据

下面介绍如何读写图像数据，内容包括以下几项。

1. 读取用多种不同标准图形文件格式保存的数据

imread 函数可读取任何受支持的图形图像文件格式和位深的图像，大部分图像文件格式使用 8 位来保存像素值。读入内存时，MATLAB 把它们保存为 unit8 型。对于支持 16 位数据的文件格式，如 PNG 和 TIFF，MATLAB 把图像保存为 unit16 类型。

注意，对于索引图像，imread 函数总是把颜色映射读入到一个 double 型数组，即使图像数组自身是 unit8 或 unit16 类型时也是如此。

例如，下面的代码将一幅 RGB 图像读入到 MATLAB 工作空间，并用变量 RGB 表示。

```
RGB = imread('football.jpg');
```

这里，imread 函数从文件的内容推测文件所使用的格式，也可以将文件格式作为一个变量指定给 imread 函数。MATLAB 支持许多通用的图形文件格式，如 BMP、GIF、JPEG、PNG 和 TIFF 等。

MATLAB 支持几种图形文件格式，如 HDF 和 TIFF，它们可以包含多幅图像，默认



时, `imread` 函数只从文件中输入第一幅图像, 要从文件中输入其他图像, 需要使用文件格式支持的语句。

例如, 使用 TIFF 文件时, 可以将一个索引值与 `imread` 函数一起使用来确定文件中希望输入的图像, 本例从一个 TIFF 文件中读取 27 幅系列图像并把图像保存到一个四维数组中, 可以用 `imfinfo` 函数确定文件中保存的图像幅数。

```
mri = unit8(zeros(128,128,1,27));  
for frame = 1:27  
    [mri(:,:,,frame),map] = imread('mri.tif',frame);  
end
```

当文件含有多幅相关图像, 比如时间序列图像时, 可以将它们保存为四维数组, 要求所有图像的大小相同。

2. 将数据写成多种不同标准图形文件格式

函数 `imwrite` 将图像按某种支持的格式写入图形文件, `imwrite` 函数的大部分语法要求把图像名和文件名作为变量, 如果文件名中包含扩展名, 则 MATLAB 根据该扩展名推测需要的文件格式。

本例从一个 Mat 文件载入索引图像 X 和它的相关颜色查找表 map, 然后把图像作为一幅位图写到文件中。

```
load clown  
Name          Size          Bytes  Class  Attributes  
X              200x320          512000  double  
caption         2x1              4      char  
map             81x3             1944   double
```

与某些图形格式一起使用 `imwrite` 函数时, 可以指定其他参数。例如, 对于 PNG 文件, 可以把位深作为其他参数。下面的例子把一幅灰度图像 I 写到一个 4 位 PNG 文件中。

```
imwrite(I,'clown.png','BitDepth',4);
```

在某些文件格式中, 二值图像可以保存为 1 位格式, 如果文件格式支持它, 默认时 MATLAB 会将二值图像写成 1 位图像, 按 1 位图像读入二值图像时, MATLAB 在工作空间中用一个 `logical` 数组表示它。

下面的例子读入一幅二值图像并将它写到一个 TIFF 文件中, 因为 TIFF 格式支持 1 位图像, 文件按 1 位格式写入磁盘。

```
BW = imread('text.png');  
imwrite(BW,'test.tif');
```

注意: 写二值文件时, MATLAB 将 `ColorType` 字段设置为 `grayscale`。
`imwrite` 函数使用如表 F3 所列规则来确定输出图像的存储类型。

表 F3 确定输出图像存储类型的规则

图像的存储类型	输出图像文件的存储类型
logical	如果指定的输出图像文件格式支持 1 位图像，imwrite 函数创建了一个 1 位图像文件，如果指定的输出图像文件格式不支持 1 位图像，如 JPEG，imwrite 函数将图像转换为 unit8 灰度图像类型
unit8	如果指定的输出图像文件格式支持 8 位图像，imwrite 函数创建了一个 8 位图像文件
unit16	如果指定的输出图像文件格式支持 16 位图像（PNG 或 TIFF），imwrite 函数创建一个 16 位的图像文件，如果指定的输出图像文件格式不支持 16 位图像，imwrite 函数将图像数据转换为 unit8 类型并创建一个 8 位图像文件
double	MATLAB 将图像数据转换为 unit8 型并创建一个 8 位图像文件，因为大部分图像文件格式使用 8 位

3. 查询图形图像文件，寻找保存在文字段中的信息

使用函数 imfinfo 可以获得图形文件的信息，获取的信息与文件类型有关，但至少包括下面的内容：

- 文件名。
- 文件格式。
- 文件格式的版本号。
- 文件修改日期。
- 文件大小，按字节计。
- 图像宽度，按像素计。
- 每像素的位数。
- 图像类型：RGB 图像、灰度图像或索引图像。

4. 转换图像的存储类型

使用 MATLAB 函数 double，可以将 unit8 和 unit16 型数据转换为 double 双精度型，但是，存储类型之间的转换改变了 MATLAB 和工具箱解释图像数据的方式，如果希望生成的数组被合理解释为图像数据，则需要再进行转换时调整 and 平衡数据。

为了便于转换存储类型，使用下面函数中的一个：im2double、im2unit8 或 im2unit16。这些函数自动控制原始数据的调整 and 平衡。下面的命令将一幅数据值在[0,1]范围内的双精度 RGB 图像转换为一幅数据大小在[0,255]范围内的 unit8 型 RGB 图像。

```
RGB3 = im2unit8(RGB1);
```

把图像转换为位数更低的类型时，通常会丢失一些图像信息。例如，一个 unit16 型灰度图像可以存储最多 65536 种不同的灰度，但 unit8 型灰度图像只能存储 256 种不同的灰度，把 unit16 型灰度图像转换为 unit8 型灰度图像时，im2unit8 函数会量子化原始图像中的灰度。



将一幅索引图像从一种存储类型转换为另一种类型并不总是可行的，在索引图像中，图像矩阵只包括索引值而不是颜色数据本身，所以在转换过程中无法进行颜色数据的量子化。

5. 转换图像的文件格式

如果想改变图像的文件格式，用 `imread` 函数读入图像，然后用 `imwrite` 函数指定合适的格式，保存图像。



第 21 例 汽车图像识别



本例以汽车为对象，在介绍图像处理工具箱中图像合成、逻辑运算及邻域和块处理基本知识的基础上，通过交通车辆图像辨识实例介绍其具体用法。通过本例学习，应掌握以下两点：

- 图像处理工具箱中图像合成、逻辑运算及邻域和块处理基本知识。
- 基于图像工具箱对汽车图像识别过程。

21.1 图像处理工具箱介绍（中）

21.1.1 图像合成

图像合成又称为图像融合，包括图像的代数运算和逻辑运算等，图像运算工具箱提供了专门的图像代数运算函数，采用 MATLAB 的逻辑操作符，可以对二值图像进行逻辑运算。

1. 代数运算

图像的代数运算基于图像的标准算术运算（加、减、乘和除等），无论是作为复杂图像处理操作的前期处理还是作为运算本身，图像代数运算都有很多用途。可以用 MATLAB 算术运算符完成图像运算，但是应用这些运算符之前，需先将图像转换为 double 型，为了使图像算术运算更方便，图像处理工具箱包含了一系列实现所有数值型非稀疏数据的处理函数，使用这些函数的优点有：

- 不需要将数据转换为 double 型。函数接受任何数值数据类型，如 unit8、unit16 和 double 等。
- 自动进行溢出控制，函数对返回值进行截断处理，使之适合相应的数据类型。

表 21-1 列出了工具箱提供的图像运算函数。



表 21-1 图像运算函数

函 数	描 述
imabsdiff	两幅图像的绝对差
imadd	两幅图像的和运算
imcomplement	图像的补运算
imdivide	两幅图像的除运算
imlincomb	计算两幅图像的线性组合
immultiply	两幅图像的积
imsubtract	两幅图像的差

注意：整型运算很容易溢出，算术运算还会生成分数值，它不能用整型数组表示。

2. 逻辑运算

对于二值图像，可以用 MATLAB 的逻辑操作符进行逻辑运算。

21.1.2 图像的空间变换

图像的空间变换，或者说几何变换，指的是通过一定的几何运算，将图像经过平移、旋转、错切、缩放等变换操作以后显示在新的位置。

1. 插值

插值用于估计图像上两个像素之间某个位置上的像素值，工具箱通过插值来获取其他像素的值，`imresize` 和 `imrotate` 函数使用了二维插值。

图像处理工具箱提供了三种插值方法：

- 最近邻插值。
- 双线性插值。
- 双三次插值。

几种插值的原理基本相同，每次处理时，首先在输入图像中找到与输出图像中对应的点，然后计算点附近某些序列的像素值的加权平均值，并将它赋给输出像素。权重由每个像素与点之间的距离确定。三种方法之间的区别主要在于点周围像素序列的取法不同。

2. 图像缩放

用 `imresize` 函数改变图像的大小，使用该函数，可以指定输出图像的大小、插值的方法和用于防止出现走样的滤波器。

使用 `imresize` 函数，可以用两种方法指定输出图像的大小：指定放大倍数和指定输出图像的尺寸。

指定放大倍数是通过指定一个大于 0 的数，可以放大或缩小图像。

指定输出图像储存指通过传递一个包含输出图像列数和行数的向量来指定输出图像的大小。

3. 旋转图像

用 `imrotate` 函数旋转图像，函数输入两个主要的变量，为旋转的图像和旋转角度。如果指定一个正值，`imrotate` 函数按逆时针方向旋转图像，如果指定一个负值，`imrotate` 函数按顺时针方向旋转图像，旋转的角度为度。

作为可选变量，还可以给 `imrotate` 函数指定插值方法和图像的大小。

指定插值方法：

- `imrotate` 函数默认用最近邻插值法确定输出图像中像素的值，但是也可以指定其他方法。其他的选择还有双线性插值法和双三次插值法。
- `imrotate` 函数默认创建一个足够包含整个原始图像的输出图像，落在原始图像边界以外的像素值设置为 0，并且在输出图像中显示为黑色的背景，如果把文本字符串“`crop`”指定为变量，则 `imrotate` 函数会将输出图像裁剪成与输入图像的大小相同。

4. 图像裁剪

用 `imcrop` 函数对图像进行裁剪，该函数输入两个变量：要裁剪的图像和定义裁剪区域的矩形坐标。

如果调用 `imcrop` 函数时没有指定裁剪矩形，可以交互式指定裁剪矩形。此时，当鼠标落在图像上方时，会变成十字形。把光标放在裁剪区域的一个角上，按下鼠标左键，然后拖拉至裁剪区域的对角。`imcrop` 函数会在所选区域的周围画一个矩形。释放鼠标时，`imcrop` 函数从选择区域创建一个新图像。

5. 一般的空间变换

用 `imtransform` 函数完成一般的二维空间变换。该函数有两个输入变量：要变换的图像和一个称为 `TFORM` 的空间变换结构，该结构指定变换类型。

在 `TFORM` 结构中指定变换类型，创建一个 `TFORM` 结构有两种方法，即使用 `maketform` 函数和 `cp2tform` 函数。

使用 `maketform` 函数时，可以指定变换类型，如表 21-2 所示。

表 21-2 变换类型

变 换	描 述
'affine'	变换，包括平移、旋转、比例、拉伸和错切等，直线仍为直线，平行线仍为平行线，但矩形可能变为平行四边形
'box'	是 affine 变换的特例
'composite'	两种或两种以上变换的组合
'custom'	自定义变换
'projective'	投影变换

一旦在 `TFORM` 结构中定义了变换，就可以通过调用 `imtransform` 函数进行变换操作。



21.1.3 邻域和块处理

1. 块处理操作

有些图像处理操作是逐块处理的，而不是一次处理整幅图像。图像处理工具箱提供了进行图像块处理的一般函数，下面介绍这些函数。

使用这些函数中的任何一个，需要提供与块大小有关的信息，并单独指定一个函数来块处理。这个单独指定的函数将输入图像分成不同的块，为每个块调用指定的函数并将结果重新分配给输出图像。

使用这些函数，可以完成不同的块处理操作，包括滑动邻域操作和分离块操作。

- 进行滑动邻域操作时，输入图像时按像素进行处理。即对于输入图像中的每个像素，进行某些操作来确定输出图像中对应像素的值，操作基于相邻像素块的值。
- 进行分离块操作时，输入图像时按块进行操作，即图像分成几个矩形块，并且有些操作是单独对每个块进行的，以便确定输出图像的对应块中像素的值。

另外，工具箱提供了进行列处理操作的函数，这些操作与块操作没有实质性的区别，不仅如此，还通过将块重置到一个矩阵列来加速块操作。

2. 滑动邻域操作

滑动邻域操作每次处理一个像素，输出图像中任何给定像素的值都通过给输入图像中对应像素邻域内像素值应用一个算法来确定，某像素的邻域是指该像素的相对位置确定的一系列像素。邻域是一个矩形块，在图像矩阵中从一个元素向下一个元素移动时，邻域块向相同方向滑动。

按照以下几个步骤进行滑动邻域操作。

- (1) 选择一个像素。
- (2) 确定这个像素的邻域。
- (3) 将一个函数应用于邻域中的像素值，这个函数必须返回一个标量。
- (4) 找到输出图像中的像素，它的位置对应于输入图像中中心像素的位置，将这个输出像素的值设为函数的返回值。
- (5) 对于输入图像中的每个像素，重复前面四个步骤。

处理这些邻域时，滑动邻域操作通常用多个 0 来填充图像边界，换句话说，这些函数通过假设图像被额外的 0 组成的行和列包围来处理边界像素，这些行和列不会成为输出图像的一部分，并且只用于图像中实际像素的邻域的一部分。

可以用滑动邻域操作实现多种滤波。实例之一是卷积，它是实现线性滤波，MATLAB 提供了 `conv` 和 `filter2` 函数，工具箱提供了 `imfilter` 函数进行卷积。

除卷积外，还有很多其他的滤波操作可以通过滑动邻域实现，这种操作实际上是非线性的。例如，可以再输出像素的值等于输入像素的邻域内像素值的标准差的地方实现滑动邻域操作。

可以用 `nlfilter` 函数实现多种滑动邻域操作，`nlfilter` 函数的输入变量有一幅图像、邻域大小和一个返回标量的函数，返回一幅大小与输入图像相同的图像。输出图像中每个



像素的值通过将对应输入像素的邻域传递给返回标量的那个函数来进行计算。

3. 分离块操作

分离块是将矩阵分成 $m \times n$ 部分的矩形分离框，分离块从图像的左上角开始无重叠地覆盖图像矩阵，如果这些块不能精确覆盖图像，则工具箱进行 0 填充。

函数 `blkproc` 进行分离块操作，该函数从图像中提取分离块并将它传递给指定的函数，然后将返回的块集中起来形成输出图像。

调用 `blkproc` 函数定义分离块时，可以指定将这些块相互重叠，即可以指定块额外的像素行和列，处理块时将它们的值考虑进去。存在重叠时，`blkproc` 函数将扩展的块传递给指定的函数。

指定重叠，需要给 `blkproc` 函数另外提供一个输入变量，重叠常常会增加所需要的 0 填充的数量。

4. 列处理

工具箱提供了多个把滑动邻域或分离块作为列进行处理的函数，它们对于那些在 MATLAB 中按列处理的操作来说很有用，很多时候，列处理可以减少图像处理的运行时间。

可以用 `colfilt` 函数进行列处理，该函数可以实现以下操作：

- 将图像矩阵的每一个滑动块或分离块重塑为一个暂时矩阵的列。
- 将这个暂时矩阵传递给一个指定函数。
- 将生成的矩阵重置为原来的形状。

21.2 基于图像处理的交通车辆辨识

本例使用 `mmreader`、`implay` 和其他图像处理函数来完成交通视频中车辆视频的检测。检测过程分为以下五个部分。

1. 通过 `mmreader` 函数读取视频文件

`mmreader` 函数能够从多媒体文件中读取视频文件并构建一个多媒体数据对象。读者可以通过帮助文档详细了解 `mmreader` 函数的用法。

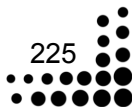
通过如下语句完成读取视频文件：

```
trafficObj = mmreader('traffic.mj2')
```

在命令行窗口中输入该句命令后出现如下提示：

```
Summary of Multimedia Reader Object for 'traffic.mj2'.
```

```
Video Parameters: 15.00 frames per second, RGB24 160x120.  
120 total video frames available.
```





通过 `get` 命令可以得到进一步关于 `trafficObj` 对象的信息：

```
get(trafficObj)

General Settings:
    Duration = 8
    Name = traffic.mj2
    Path = C:\Program Files\MATLAB\R2010b\toolbox\images\indemos
    Tag =
    Type = VideoReader
    UserData = []

Video Settings:
    BitsPerPixel = 24
    FrameRate = 15
    Height = 120
    NumberOfFrames = 120
    VideoFormat = RGB24
    Width = 160
```

2. 通过 `implay` 命令显示图像

在命令行中输入：

```
implay('traffic.mj2');
```

则出现交通视频截图如图 21-1 所示。



图 21-1 交通视频截图

3. 设计自己的算法

当处理视频文件时，通常的做法是选择视频中具有代表性的帧文件，并基于该帧文

件设计自己的算法。这样设计的算法就能够应用到视频中的所有帧上。

对于车辆辨识来说，在实际的交通视频文件中一般既有浅色的车辆，又有深色的车辆。这给车辆辨识带来了困难，常用的做法是首先对图像进行简化，比如说如果对浅色车辆感兴趣，则可以将所有非浅色车辆剔除出去，这样就大大简化了后续辨识的工作量。

一种从视频文件中去除深色车辆的方法是使用 `imextendedmax` 函数。这个函数能够返回一个二值图像。该二值图像只能显示黑或者白，选择黑或者白是根据原图像数值的大小是否超过指定阈值进行判断，如果超过则显示为白，如果未超过则显示为黑。经过这个函数，除深色车辆外其他部分都变为背景，而图中的白色区域则为深色车辆。如果原来图像是 RGB 图像，则可以通过 `rgb2gray` 函数变换为灰度图像。

下面的程序给出辨识深色车辆的例子：

```
darkCarValue = 50;
darkCar = rgb2gray(read(trafficObj,71));
noDarkCar = imextendedmax(darkCar, darkCarValue);
imshow(darkCar)
figure, imshow(noDarkCar)
```

在该程序中将深色车辆的阈值(`darkCarValue`)设为 50，原图是 RGB 图，通过 `rgb2gray` 将其变换为灰度图像。

辨识前的图像如图 21-2 所示，辨识后的图像如图 21-3 所示。



图 21-2 深色车辆辨识前图像

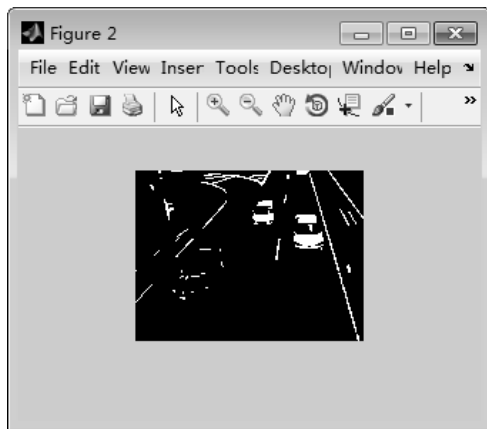


图 21-3 深色车辆辨识后图像

在处理后的图像中可以看出，除深色车辆外，还有许多其他异物存在，如车道标记等。需要通过一定的方法将非深色车辆部分去除掉。通过区域最大值处理方式是不会删除车道标线的，因为它们的像素值均高于阈值。要删除这些对象，可以使用函数 `imopen`。该函数使用形态学处理方式，从一个二进制图像中除去小的物体，同时保持大的对象。使用形态学处理时，必须决定在操作中使用的结构元素的大小和形状。因为车道标记长而薄，可以在像素区域使用 `implay` 工具来估计这些对象的宽度。

下面程序给出进一步去除非深色车辆对象的例子：



```
sedisk = strel('disk',2);  
noSmallStructures = imopen(noDarkCar, sedisk);  
imshow(noSmallStructures)
```

经过 `imopen` 函数处理后的图像如图 21-4 所示。从图中可以看出,深色车辆周围的车道标识都已经去掉了。

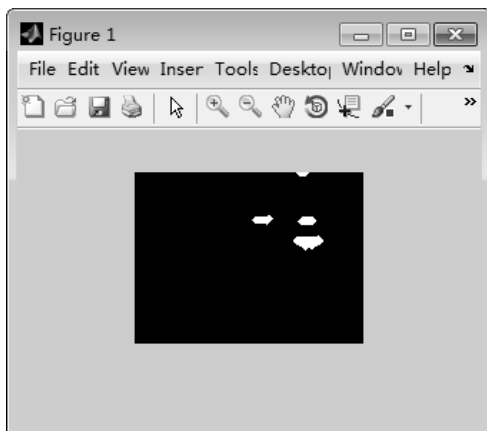


图 21-4 经过 `imopen` 处理的图像

4. 将算法应用到视频处理中

之前的程序只能够处理一帧图像文件,为处理整个视频文件,需要设置一个循环,逐帧逐帧的处理整个视频文件,从而达到处理整个视频的目的。有的视频文件较大,如果需要一次完成整个视频文件的处理,则需要较大的内存。

下面给出处理整个视频文件的例子:

```
nframes = get(trafficObj, 'NumberOfFrames');  
I = read(trafficObj, 1);  
taggedCars = zeros([size(I,1) size(I,2) 3 nframes], class(I));  
  
for k = 1 : nframes  
    singleFrame = read(trafficObj, k);  
  
    %转换成灰度图像  
    I = rgb2gray(singleFrame);  
  
    % 去除深色车辆  
    noDarkCars = imextendedmax(I, darkCarValue);  
  
    % 去除车辆标识线  
    noSmallStructures = imopen(noDarkCars, sedisk);  
  
    % 去除小结构物体  
    noSmallStructures = bwareaopen(noSmallStructures, 150);
```



```
% 获取在该帧的其余的每个对象的面积和形心。
% 最大面积的对象是浅色的汽车。
% 创建一个原始帧的副本并通过改变标记车的重心像素为红色。
taggedCars(:, :, :, k) = singleFrame;

stats = regionprops(noSmallStructures, {'Centroid', 'Area'});
if ~isempty([stats.Area])
    areaArray = [stats.Area];
    [junk, idx] = max(areaArray);
    c = stats(idx).Centroid;
    c = floor(fliplr(c));
    width = 2;
    row = c(1)-width:c(1)+width;
    col = c(2)-width:c(2)+width;
    taggedCars(row, col, 1, k) = 255;
    taggedCars(row, col, 2, k) = 0;
    taggedCars(row, col, 3, k) = 0;
end
end
```

5. 观测标识结果

下面的程序给出辨识结果，图像如图 21-5 所示。

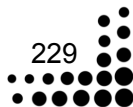
```
frameRate = get(trafficObj, 'FrameRate');
implay(taggedCars, frameRate);
```



图 21-5 车辆辨识结果

21.3 本例小结

本例首先介绍了图像处理工具箱中图像合成、逻辑运算及邻域和块处理的基本知识，其次通过交通车辆图像辨识实例介绍其图像处理工具箱在交通视频车辆识别中的应用。





第 22 例 飞机航拍图处理



本例以飞机为对象，在介绍图像处理工具箱中图像分析和图像匹配基本知识的基础上，通过飞机航拍图处理实例介绍其具体用法。通过本例学习，应掌握以下两点：

- 图像处理工具箱中图像分析和图像匹配基本知识。
- 基于图像工具箱对飞机航拍图处理过程。

22.1 图像处理工具箱介绍（下）

22.1.1 图像分析

MATLAB 图像处理工具箱提供了灰度统计、直方图、等值线图和边缘检测、边界跟踪四叉树分解等图像分析功能。

图像处理工具箱提供了几个函数来返回与图像数据有关的信息，这些函数以不同形式返回这些信息，包括：

- 选定点的数据值。
- 图像中沿一定路径分布的数据值。
- 图像数据的等值线图。
- 图像数据的直方图。
- 图像数据的综述统计量。
- 图像区域的特征度量。

1. 像素选择

工具箱中有两个函数提供与图像中指定像素的颜色数据值有关的信息。

- `pixval` 函数可以交互显示图像上像素的数据值，它还可以显示两个像素之间的欧拉距离。
- `impxel` 函数返回选定的一个或一系列像素的数据值，可以将选定像素的坐标作为输入变量，或者可以用鼠标选择像素。

注意：对于索引图像，`pixval` 函数和 `impxel` 函数都显示保存在颜色映射矩阵中的 RGB 值，而不是索引值。



使用 `pixval` 函数时, 首先显示一幅图像, 然后输入 `pixval` 命令, `pixval` 函数在图像下面放一个黑条, 这个黑条可以显示鼠标下方像素的坐标值和该像素的颜色数据。

如果鼠标在图像上单击以后进行拖拉, 则 `pixval` 函数还会显示单击点与鼠标当前点之间的欧拉距离, `pixval` 通过在这些点之间画线来表示它们之间的距离已经测量过了, 释放鼠标时, 直线和距离显示消失。

`impixel` 函数即时给出的结果比 `pixval` 函数给出的少, 但使用 `impixel` 函数有用变量返回结果的好处, 这个变量可以以交互或非交互的方式被调用, 如果以无参数方式调用 `impixel` 函数, 则鼠标位于图像上方时光标变成十字形。然后可以单击感兴趣的像素, `impixel` 函数在选择的每个像素上显示一个小的星形, 选定以后, 单击回车键, `impixel` 函数返回选定像素的颜色值, 然后星形消失。

2. 灰度轮廓

`improfile` 函数沿图像的线段或多义线计算和显示灰度, 可以将直线段的端点坐标作为输入变量, 也可以用鼠标定义路径。使用第 2 种方法时, `improfile` 函数使用插值法来确定路径上等间隔点的值, `improfile` 函数用于灰度图像和 RGB 图像时效果最佳。

如果只有一条直线段, 则 `improfile` 函数用二维图显示灰度值, 如果是多义线, 则 `improfile` 函数用三维图显示灰度值。

如果调用无参数的 `improfile` 函数, 则鼠标光标位于图像上方时变为十字形, 然后通过单击线段的终点来定义线段, `improfile` 函数会在两个连续选定的点之间进行连线, 指定完路径以后, 单击回车键, `improfile` 函数将图形显示在一个新的图形窗口中。

3. 图形等值线

可以用 `imcontour` 函数显示灰度图中数据的等值线图, 该函数与 MATLAB 中的 `contour` 函数相似, 但会自动设置坐标轴, 使得它们的方向和大小比例与图像匹配。

4. 图像直方图

图像等值图是一种显示索引图像或灰度图像亮度分布的图形, 用 `imhist` 函数创建图像直方图, 该图首先将数据分成 n 个等间距的条形, 每个条形表示一个数据范围, 然后计算落在这个范围内像素的个数。

5. 综述统计量

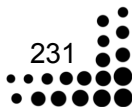
可以用 `mean2`、`std2` 和 `corr2` 函数计算图像的标准统计量, `mean2` 和 `std2` 函数计算矩阵中元素的均值和标准差, `corr2` 函数计算两个大小相同的矩阵的相关系数。

6. 区域属性度量

可以用 `regionprops` 函数计算图像区域的属性, 如度量面积、质心和包围盒等属性。

7. 边缘检测

可以用 `edge` 函数进行边缘检测, 这里的边缘指的是图像中对象的边界, 为了进行边





缘检测，edge 函数使用下面两个准则中的一种，在图像上查找亮度剧烈改变的地方。

- 亮度的一阶导数在量级上比某些阈值更大的地方。
- 亮度的二阶导数为 0 的地方。

edge 函数提供了多个求导器，每个求导器可以解决上面问题中的一种，对于有些求导器，可以指定操作是否对水平边界、垂直边界或两者都敏感。edge 函数返回一个二值图像，图像中找到了边界的像素用 1 表示，否则用 0 表示。

edge 函数提供的边缘检测方法中，最有力的是 Canny 法，该方法与其他边缘检测方法的主要区别在于，它使用两个阈值来检测强边界和弱边界，并且只在弱边界与强边界相连时才进行显示，所以该方法与其他方法相比，受噪声影响的机会更小，并且更有可能检测到真实的弱边界。

8. 边界跟踪

工具箱提供了函数，可以利用它们查找二值图像中的对象边界，这两个函数是 bwtraceboundary 函数和 bwboundaries 函数。

bwtraceboundary 函数图像中对象边界上所有像素的行坐标和列坐标，必须指定对象上一个边界像素的位置作为跟踪的起点，bwboundaries 函数返回图像中所有对象边界像素的行坐标和列坐标。对这两个函数，非 0 像素属于对象，值为 0 的像素组成背景。

9. 四叉树分解

四叉树分解是一种图像分析技术，它将图像二次分解为比图像本身更均一的块，该技术揭示了图像的结构信息，它还可以用于适应性压缩算法的第 1 步。

可以用 qtdecomp 函数进行四叉树分解，该函数将一个方形图像分成 4 个大小相同的块，然后测试每一个块，看它是否符合某些均一性准则，如果符合准则，则不做进一步的分解，如果不符合准则，将该块再分解为 4 个小块，然后测试其他块，重复这个过程，直到每个块都符合准则，结果可能有几种不同大小的块。

22.1.2 图像配准

图像配准指的是将同一场景的两幅或多幅图像进行对准。一个典型的应用是，将一幅图像作为其他图像的参照进行比较，图像配准的目的是通过输入图像进行空间变换，使输入图像与基准图像对准。

空间变换是将一幅图像中的位置映射到另一幅图像中的新位置，确定使图像对准的变换参数对于图像配准来说很关键。

图像配准常常用作其他图像处理应用的前处理步骤。例如，可以用图像配准来对准地表卫星图像或核磁共振图像。配准以后，可以对图像进行比较，看河流如何迁移，大地如何泛滥。

配准图像的一般过程有以下几步。

1. 点映射

图像处理工具箱提供了一些支持点映射的工具，利用它们，可以确定使图像和其他图像配准的变换参数，进行点映射时，在成对图像中选择点来确认图像中的相同特征和标志，然后，根据这些控制点的位置来推导出某些空间映射关系。

使用点映射的图像配准包括以下几个步骤。

- (1) 将图像读入到 MATLAB 工作空间。
- (2) 指定图像中的成对控制点。
- (3) 保存控制点对。
- (4) 用反相关调整控制点。
- (5) 指定要使用的变换类型，并根据控制点对推测参数。
- (6) 对没有配准的图像进行变换，使之对准。

2. 支持的变换类型

cp2tform 函数可以计算六种类型的变换，表 22-1 中按照复杂度列出了这些变换。

表 22-1 变换类型及描述

变 换 类 型	描 述	最小控制点对数
'Linear conformal'	当输入图像中的形状没有改变，但是图像经过平移、旋转和比例等的组合变换以后发生失真时使用本变换。变换以后，直线仍然是直线，平行线仍然是平行线	2 对
'affine'	当输入图像中的形状出现错切现象时使用本变换，直线仍然是直线，平行线仍然是平行线，但是矩形变成了平行四边形	3 对
'projective'	景物显得倾斜时使用本变换，变换后直线仍然是直线，但平行线不再平行	4 对
'polynomial'	图像中的对象发生弯曲时使用本变换，多项式的阶数越高，拟合效果越好，但是生成的图像比基准图像包含更多的曲线	6 对（2 阶） 10 对（3 阶） 16 对（4 阶）
'piecewise linear'	当图像中的变形现象具有分段性时使用本变换	4 对
'lwm'	当变形有局部性的变化，而且分段线性的条件不够充分时使用本变换	6 对（推荐 12 对）

前四种变换为全局变换，在这些变换中，单一的数学表达式适用于整幅图像，最后两种变换，是局部变换，在这些变换中，不同的数学表达式适应于图像中不同的区域。



3. 选择控制点

在要配准的成对图像中指定控制点，需要使用控制点选择工具 `cpselect`，该工具在基准图像后面显示要配准的图像，即输入图像。按下面四步指定控制点。

(1) 启动工具，指定输入图像和基准图像。

(2) 查看图像，寻找在两幅图像中都可辨认的可视元素，`cpselect` 函数提供了多种方法来查看图像，包括平移和缩放等。

(3) 指定两幅图像中的匹配控制点。

(4) 将控制点保存到 MATLAB 工作空间中。

4. 保存控制点

指定控制点对以后，必须将它们保存到 MATLAB 工作空间中，以便在后面的图像配准中使用它们，按照以下步骤保存控制点。

(1) 在控制点选择工具中选择“File”菜单。

(2) 选择“Save Points to Workspace”选项。

22.2 大型飞机航拍图处理

函数 `blockproc` 非常适合将对图像的操作应用于图像的分块对象上，然后将分块图像处理的结果组合在一起，并将它们组合在一起形成新的图像。但是许多图像处理算法，需要“全局性”的信息，即需要全部图像的数据。这在图像较小的时候可以实现，但是实际图像有时候很大，如飞机航拍图像或者卫星扫描形成的图像。这些图像文件很大，如果每次都需要针对整体图像进行处理，则计算效率会比较低下，这就给图像处理带来了困难。本例探讨如何在使用 `blockproc` 函数进行分块图像处理的基础上，统计整体图像的信息，然后再将这些信息更准确地应用到处理分块图像中去。

例子分为五个步骤，下面分别予以介绍。

1. 生成一个真彩色复合图像

通过 `blockproc` 函数从“`rio.lan`”中读取数据文件，“`rio.lan`”是一个以 Erdas LAN 格式存储的陆地图像文件，如图 22-1 所示。`blockproc` 函数只能够读取 TIFF 或者 JPEG2000 格式的文件，如果要读取其他类型的文件，读者需要编写图像适配接口来完成读取。在本例中，使用了一个图像读取类“`LanAdapter`”来完成读取工作。如果读者对读取图像接口文件感兴趣，可参见相关帮助文档。

Erdas LAN 格式的文件包含可见光中红、绿和蓝色光谱的宽度分别为 3、2 和 1。通过 `blockproc` 函数将这些可色光谱分解出来并形成 RGB 图像。

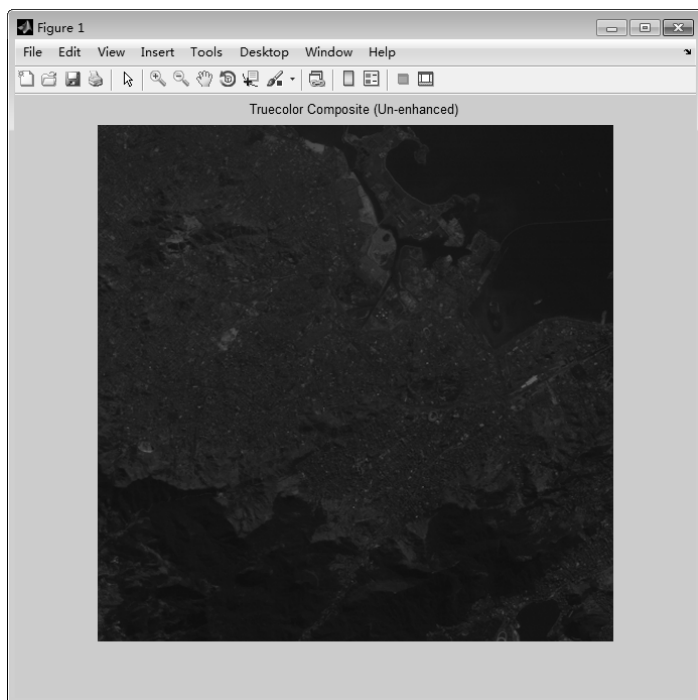


图 22-1 初始真彩图像

该部分程序如下：

```
% 生成 rio.lan 的 LanAdapter 对象
input_adapter = LanAdapter('rio.lan');

% 选择、绿和蓝光谱
input_adapter.SelectedBands = [3 2 1];

% 生成一个分块函数来返回未改动的分块数据
identityFcn = @(block_struct) block_struct.data;

% 生成刚开始的真彩图像
truecolor = blockproc(input_adapter,[100 100],identityFcn);

% 显示未改动的图像
figure;
imshow(truecolor);
title('Truecolor Composite (Un-enhanced)');
```

2. 第一次增强图像

首先通过 blockproc 函数来伸展整个动态范围的数据。这次变换定义一个函数句柄来调用每块图像的数据。

该部分程序如下：



```
adjustFcn = @(block_struct) imadjust(block_struct.data,...  
    stretchlim(block_struct.data));  
truecolor_enhanced = blockproc(input_adapter,[100 100],adjustFcn);  
figure  
imshow(truecolor_enhanced)  
title('Truecolor Composite with Blockwise Contrast Stretch')
```

图 22-2 所示为第一次增强后的真彩图像，从图中可以看出，第一次增强后的结果是不正确的。存在的问题是 stretchlim 函数计算所述输入图像的直方图，并使用该信息来计算的拉伸限制。由于每个块的调整，从它邻块的隔离，每个块计算从它局部直方图不同的限制。

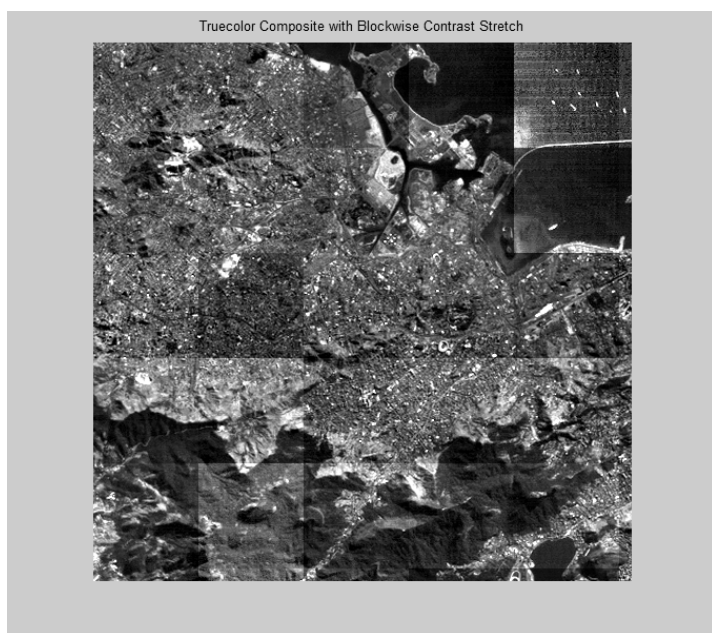


图 22-2 第一次增强后的真彩图像

3. 检查直方图累加类

为了检验整个动态范围图像的数据分布，可以计算三个可见光波段的直方图。

当处理一个足够大的图像时，不能简单地调用 imhist 创建一个图像的直方图。一个逐步建立直方图的方法是，使用 blockproc 函数将每块图像直方图累计起来。

输入下述命令来检查 HistogramAccumulator 类：

```
type HistogramAccumulator
```

在命令窗口中将会显示：

```
% HistogramAccumulator Accumulate incremental histogram.  
% HistogramAccumulator is a class that incrementally builds up a  
% histogram for an image. This class is appropriate for use with 8-bit
```



```
% or 16-bit integer images and is for educational purposes ONLY.

% Copyright 2009 The MathWorks, Inc.
% $Revision: 1.1.6.1 $ $Date: 2009/11/09 16:24:41 $

classdef HistogramAccumulator < handle

    properties
        Histogram
        Range
    end

    methods

        function obj = HistogramAccumulator()
            obj.Range = [];
            obj.Histogram = [];
        end

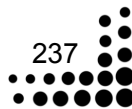
        function addToHistogram(obj,new_data)
            if isempty(obj.Histogram)
                obj.Range = double(0:intmax(class(new_data)));
                obj.Histogram = hist(double(new_data(:)),obj.Range);
            else
                new_hist = hist(double(new_data(:)),obj.Range);
                obj.Histogram = obj.Histogram + new_hist;
            end
        end
    end
end
```

这个 HistogramAccumulator 类能够将直方图进行简单的封装，允许将数据添加到一个直方图中。它并不只适用于 blockproc 函数。下面给出使用 HistogramAccumulator 类的例子，使用结果如图 22-3 所示。

```
% 生成 HistogramAccumulator 类
hist_obj = HistogramAccumulator();

% 将样本图像分割为两半
full_image = imread('liftingbody.png');
top_half = full_image(1:256,:);
bottom_half = full_image(257:end,:);

% 计算直方图增量
hist_obj.addToHistogram(top_half);
hist_obj.addToHistogram(bottom_half);
computed_histogram = hist_obj.Histogram;
```





```
% 与函数 IMHIST 计算的结果进行对比
normal_histogram = imhist(full_image);

% 检查结果
figure
subplot(1,2,1);
stem(computed_histogram,'Marker','none');
title('Incrementally Computed Histogram');
subplot(1,2,2);
stem(normal_histogram,'Marker','none');
title('IMHIST Histogram');
```

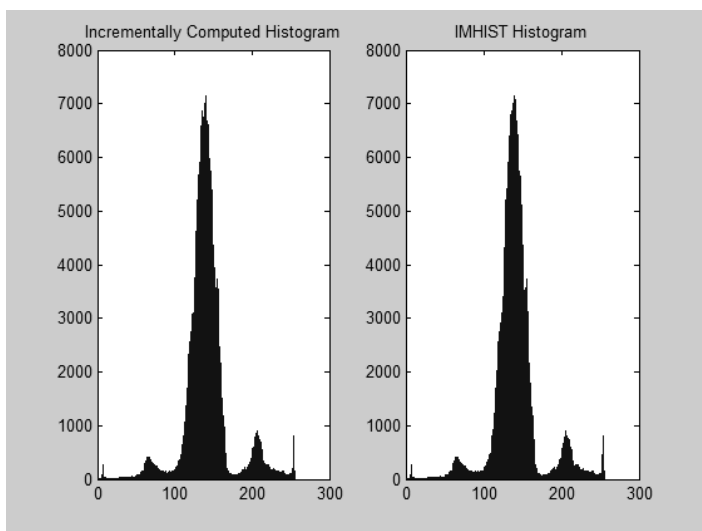


图 22-3 HistogramAccumulator 类使用例子

4. 使用 HistogramAccumulator 类和 blockproc 函数

使用 HistogramAccumulator 类与 blockproc 函数的红色条带的数据在 rio.lan 中建立直方图。可以定义一个函数句柄来对每个数据块的 blockproc 调用 addToHistogram 方法。图 22-4 所示为红色谱的直方图。通过查看这个直方图，可以看到数据可用的动态范围集中在一小部分。其他可见光波段具有相似的分布。这是原来显示的复合真彩图沉闷的原因之一。

```
% 生成 HistogramAccumulator 类
hist_obj = HistogramAccumulator();

% 设置 blockproc 函数句柄
addToHistFcn = @(block_struct) hist_obj.addToHistogram(block_struct.
data);

% 计算 histogram 的红色谱
```



```
% 注意到 addToHistFcn 函数句柄没有生成输出，
% 这是因为传递给 blockproc 的句柄没有返回
input_adapter.SelectedBands = 3;
blockproc(input_adapter,[100 100],addToHistFcn);
red_hist = hist_obj.Histogram;

% 显示结果
figure
stem(red_hist,'Marker','none');
title('Histogram of Red Band (Band 3)');
```

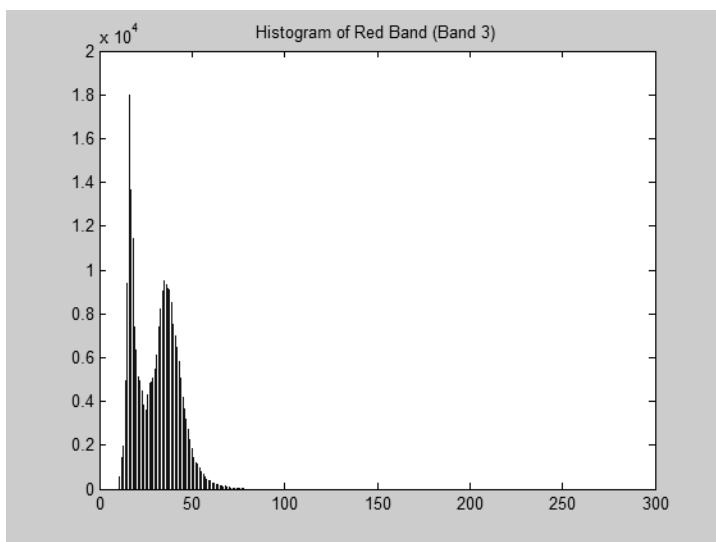


图 22-4 红色谱的直方图

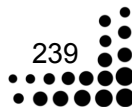
5. 通过对比拉伸来增强复合真彩图

此时可以对图像执行正确的对比拉伸。对于传统的，在内存中工作流程，可以简单地使用 `strethlim` 函数来计算 `imajust` 的参数。正如所看到的，`strethlim` 函数不适用于 `blockproc` 函数，因为该函数依赖于完整的图像信息。

一旦计算出每个可见光段图像直方图，可以手动计算出 `imajust` 的参数。

首先计算绿和蓝光谱：

```
%计算绿色光谱直方图
hist_obj = HistogramAccumulator();
addToHistFcn = @(block_struct) hist_obj.addToHistogram(block_struct.
data);
input_adapter.SelectedBands = 2;
blockproc(input_adapter,[100 100],addToHistFcn);
green_hist = hist_obj.Histogram;
```





```
% 计算蓝色光谱直方图
hist_obj = HistogramAccumulator();
addToHistFcn = @(block_struct) hist_obj.addToHistogram(block_struct.
data);
input_adapter.SelectedBands = 1;
blockproc(input_adapter,[100 100],addToHistFcn);
blue_hist = hist_obj.Histogram;
```

绿和蓝光谱如图 22-5 所示。

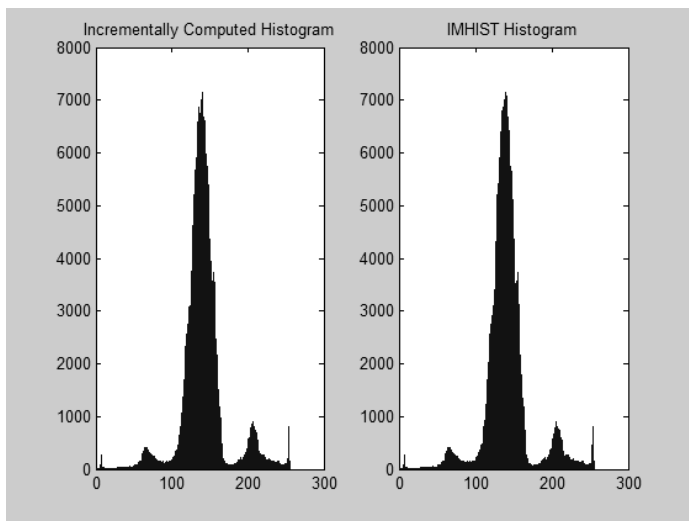


图 22-5 绿和蓝光谱

现在计算每个直方图的 CDF，并准备调用 `imajust` 函数：

```
computeCDF = @(histogram) cumsum(histogram) / sum(histogram);
findLowerLimit = @(cdf) find(cdf > 0.01, 1, 'first');
findUpperLimit = @(cdf) find(cdf >= 0.99, 1, 'first');

red_cdf = computeCDF(red_hist);
red_limits(1) = findLowerLimit(red_cdf);
red_limits(2) = findUpperLimit(red_cdf);

green_cdf = computeCDF(green_hist);
green_limits(1) = findLowerLimit(green_cdf);
green_limits(2) = findUpperLimit(green_cdf);

blue_cdf = computeCDF(blue_hist);
blue_limits(1) = findLowerLimit(blue_cdf);
blue_limits(2) = findUpperLimit(blue_cdf);

% Prepare argument for IMADJUST.
rgb_limits = [red_limits' green_limits' blue_limits'];

% 投影到[0,1]范围
```




```
rgb_limits = (rgb_limits - 1) / (255);

% 生成一个新的适应函数，用于将全局拉伸限制应用到真彩图像中
adjustFcn = @(block_struct) imadjust(block_struct.data,rgb_limits);

% 选择全 RGB 数据
input_adapter.SelectedBands = [3 2 1];
truecolor_enhanced = blockproc(input_adapter,[100 100],adjustFcn);
figure;
imshow(truecolor_enhanced)
title('Truecolor Composite with Corrected Contrast Stretch')
```

修正后真彩图如图 22-6 所示，从图像可以看出，生成的图像已经大大改善了，覆盖了更多的动态范围，并通过使用 blockproc 函数来避免将整个图像加载到内存中。

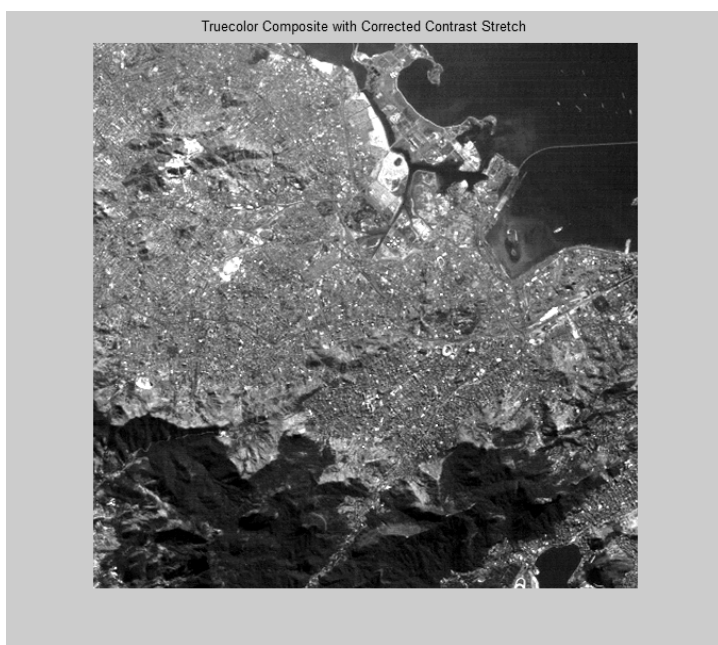


图 22-6 修正后真彩图

22.3 本例小结

本例首先介绍了图像处理工具箱中图像图像分析和图像匹配的基本知识，其次通过飞机航拍图处理实例介绍其图像处理工具箱在图像光谱分析等方面的应用。



第 23 例 船舶定位研究

本例以船舶为对象，在介绍地图工具箱中创建地图和地理计算基本知识的基础上，通过船舶定位实例介绍其具体用法。通过本例学习，应掌握以下两点：

- 地图工具箱中创建地图和地理计算基本知识。
- 基于地图工具箱进行定位过程。

23.1 地图工具箱介绍（上）

使用地图工具箱，可以在 MATLAB 中读取、分析和显示地理信息。因为地球和大部分天体通常都是球形的，所以地理数据常常在球坐标系或椭球坐标系中定义，地球曲面上定义的距离、方位、面积甚至直线都与 MATLAB 笛卡儿坐标系中的不同。

将球体上的地理信息显示到平面上还需要特殊的绘图技巧，地图工具箱可以用简单的命令创建地图，使用工具箱提供的地图数据，可以创建详细的底图，在底图上可以绘制自己的结果，还可以导入高精度的地图数据，这些数据可以从政府或研究性网站上得到。

23.1.1 创建地图

1. 创建底图

地理数据通常显示在底图上，底图中包含有类似海岸线或地形线的基本特征信息。创建底图的第 1 步是选择合适的投影方法，投影指的是如何将球体表面的信息表示在平面上。迄今为止，已有多种不同的投影方法，每一种都在比例、形状和方向的保真性方面进行折中处理。选择投影方法应该适合当前任务。准备工作的第 2 步是选择和显示合适而且详细的地图数据，数据的详细程度和数量要与底图的覆盖范围相吻合。

地图工具箱创建地球底图的命令是 `worldmap`，该命令选择一种投影方法和适合指定区域的地图集部分数据，可以用某个洲或国家的名称，或纬度和经度范围定义区域，如果对名称的拼写不知道，使用没有输入变量的 `worldmap` 命令，从显示的列表中选择名称。

下面给出南美洲地图制作示例：

首先创建底图，如图 23-1 所示。其次在底图上添加数据，其结果如图 23-2 所示。

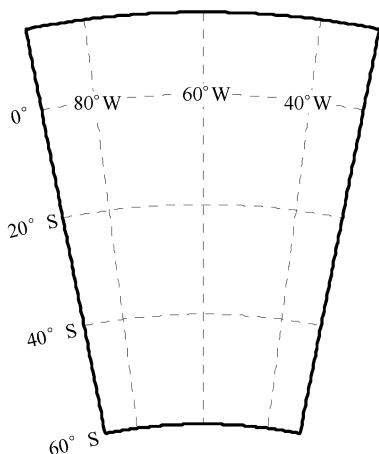


图 23-1 南美洲底图

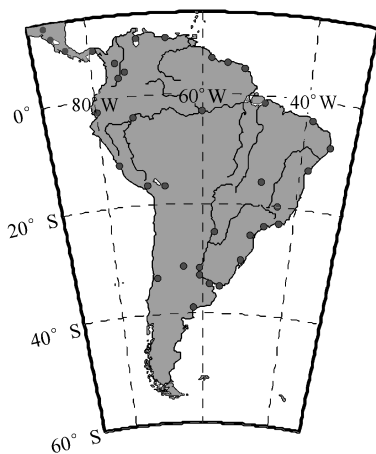


图 23-2 南美洲轮廓及主要河流城市信息

```
figure
worldmap 'south america' % 创建地图
axis off
%在底图上添加数据
geoshow('landareas.shp', 'FaceColor', [0.5 0.7 0.5]) % 陆地形状数据
geoshow('worldrivers.shp', 'Color', 'blue') % 主要河流数据
geoshow('worldcities.shp', 'Marker', '.', 'Color', 'red') % 主要城市数据
```

2. 在底图上显示数据

地图工具箱可以显示向量、矩阵和结构类型的地理数据。工具箱中的命令名通常与 MATLAB 中的图形命令相似，只是在末尾添加一个“m”，这个“m”表示数据是地理数据，并且要用已经定义的地图投影进行显示。例如，在 MATLAB 中用 `plot(x,y)` 将一个点向量显示为直线段，相应地对于地理数据使用 `plotm(lat,long)` 的语法格式。

因为大部分地图中间都包含了几种信息，新的绘图命令向地图坐标系中添加图形元素而不是替换图形元素。另外，x 坐标和 y 坐标的比例设置为相等，用 `daspectm` 命令控制 z 轴的比例。

显示矩阵数据的一个通用办法是显示为等值线图，地图工具箱提供了多种方法来表示等值线，等值线用 `contourm` 命令创建，填充的等值线用 `contourfm` 命令创建，将矩阵显示为平面并用 `contourmap` 命令控制颜色查找表可以达到使用 `contourfm` 命令绘制等间隔的等值线的相同效果。

下面以朝鲜半岛为例介绍绘制等高线过程，其绘制结果如图 23-3 所示。

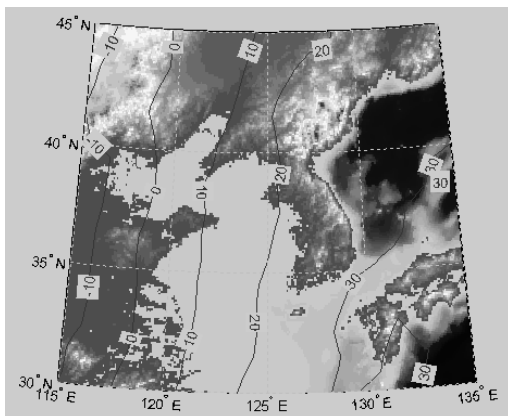


图 23-3 朝鲜半岛等高线图

```
load korea
S = shaperead('landareas', 'UseGeoCoords', true);
figure;
worldmap(map, refvec)
geoshow(map, refvec, 'DisplayType', 'texturemap');
colormap(demcmap(map))
axis off
load geoid
[c,h] = contourm(geoid,geoidlegend,-100:10:100,'r');
ht = clabelm(c,h);
set(ht,'color','r')
hidem(gca)
```

3. 导入高分辨率地图集数据

除了工具箱提供的地图集数据外，还可以导入很多高分辨率数据集，这些数据集的水平分辨率为 10km~100m。支持的向量数据包括详细的向量海岸线、世界数字图形和它的继承者 VMAP0，可以用命令行函数和图形用户界面导入数据。

23.1.2 地理计算

除了地理显示能力外，工具箱还包括一个用于分析地理数据的扩展命令集。包括单位转换、地理几何（如距离、方位、高程、定位和相交）等方面的命令。也可以创建地理数据（如跟踪线、圆、椭圆和缓冲区）等。

1. 地理空间数据

地图可以简单地定义为地理数据的图形表示。地图工具箱中，地图数据是表示地理位置、区域属性或星球表面特征的任何变量或变量集，数据的大小、复杂度和格式没有限制。使用工具箱提供的函数和图形用户界面，可以将这些数据通过多种方法渲染成地图。



地理空间数据取自多种形式和格式的数据，它的结构比非地理数据更复杂，实际上，它是空间数据的一个子集，只表示点在给定坐标系中的位置。

2. 地图数据

目前用于表示地形的地理空间数据类型主要有向量数据和栅格数据两种。向量数据用点、直线和多边形表示对象的形状，而栅格数据把空间分割成用值表示的单元。可以将这两种数据组合起来使用，在栅格数据上叠加向量数据进行显示。

3. 向量数据

向量数据可以表示地图，这些向量用经度-纬度或投影坐标对表示点集、线性地图或区域地图的特征。使用这种表示方法时，地理数据以向量格式保存，绘成的地图称为向量地图，这些数据由指定的坐标点列组成，并用其他数据表示与相邻点的连接关系。

地图工具箱中，向量数据由先后连接的地理或投影坐标对组成，数据的分隔可以用分隔符 NaNs 表示，也可以创建单独的向量变量，对于向量地图数据，数据的连通性通常只影响图形显示，但它对于地图的统计计算也有影响。

4. 栅格数据

还可以将矩阵数据用地图表示，这个矩阵中每个行与列交叉点上的元素对应于特定地理区域的矩形面片，另有数据表示它与相邻面片的连通性。这通常称为栅格数据，当栅格格式的数据表示星球表面时，称为数据网格，数据保存为数组或矩阵。MATLAB 的矩阵操作函数完全适用于这类数据，栅格的值可用单元内的平均值或中心值表示。

当栅格数据由表面高程组成时，地图可称为数字高程模型（DEM），它显示为地形图。DEM 是数字地形模型（DTM）最通用的形式，还可以用等值线、三角高程点、四叉树、八叉树等表示。工具箱中的 topo 世界地形数据就是 DEM 的一个例子。

栅格数据还包括地理引用图像数据，与数据网格类似，图像数据用行和列表示。但它们之间有一些小小的差别，理解这些差别很重要。差别之一是，图像可能将 RGB 值或多谱信道保存在一个单独的数组中，这样它就有了第三维。另一个差别是，当数据网格保存为 double 型时，图像可能使用 MATLAB 保存类型的数据范围，最通用的是 unit8、unit16、double 和 logical 等。第三个差别是，对于 double 型的灰度图像或 RGB 图像，数组元素中的值限制在[0 1]内部。

向量变量和数据网格变量常常一起使用或显示。例如，向量形式的大陆架可能与一个温度数据网格一起使用，并使后者更有用。当多个地图变量不管变量类型一起使用时，可以认为是一套单独的地图数据。当然，要做到这一点，不同的数据集必须使用相同的坐标系。

以 coast 和 topo 数据集为例，介绍向量变量和数据网格变量使用。

（1）清除当前地图。

```
clma
```

（2）重新载入海岸线数据。



```
load coast
```

(3) 如果 topo 数据在工作空间中不存在，也载入它。

```
load topo
```

(4) 设置 Robinson 投影。

```
axesm robinson
```

(5) 用合适的颜色查找表绘制栅格地形数据的图。

```
geoshow(topo,topolegend,'DisplayType','texturemap')  
demcmmap(topo)
```

(6) 在地形图上方绘制海岸线数据的图。

```
geoshow(lat,long,'Color','r')
```

注意：可以用 geoshow 函数同时显示栅格数据和向量数据，生成的地图如图 23-4 所示。

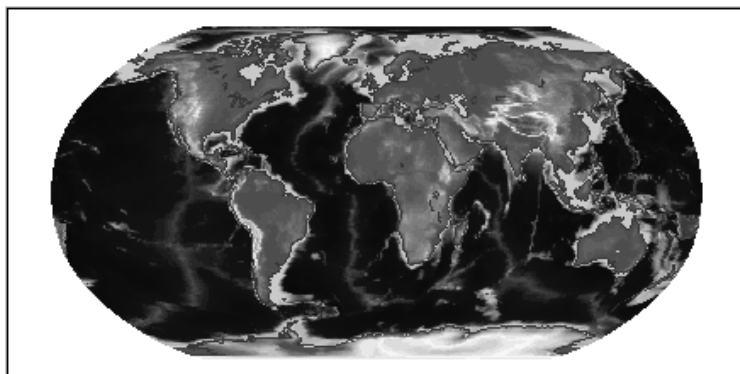


图 23-4 同时显示栅格数据和向量数据

5. 操作向量数据

使用地图工具箱可以用不同的方式操作、组合和区分向量地理数据。

6. 重新组装向量对象

当部分直线和面片被组合到元素中并用 NaNs 分隔到很大的向量中以后就很难识别，可以用 polysplit 函数把这些多边形或直线向量分解成基本图形单元，该函数将列向量作为输入变量。

7. 匹配直线段

将具有公共端点的直线段进行连接是与直线段有关的一个通用操作。polymerge 命令比较保存在纬度数据向量和经度数据向量中的线段端点数据，以便识别精确匹配或距离



在一定范围内的端点，然后将匹配的直线段进行连接。重复此过程，直到没有公用点可以找到为止，有两个必需的变量是纬度向量和经度向量。

8. 地理插值

使用向量数据，作为点间地理真实性有关的假设时必须小心。例如，用向量数据绘图时，可能会用直线段连接两个点，通常这并不表示这些点之间的任何真实信息，沿海岸线的点的分布可能比较稀疏，在缺少其他信息的情况下，在两点之间插入其他点可能会导致错误。

9. 向量相交

地图工具箱提供了一系列函数对向量数据进行求交计算，这些函数还可以计算任意向量数据的交。

10. 多边形面积

可以用函数 `areaInt` 计算多边形格式向量数据的地理面积。

11. 通过布尔操作叠加多边形

多边形布尔操作可以解决一系列与向量数据多边形对象逻辑关系有关的问题，标准布尔操作包括交、并、差或异或等。`polybool` 函数可以对两个向量集合进行这些操作。

12. 操作栅格数据

有一些操作如属性的逻辑操作、表面特性的计算等使用栅格数据更合适。

13. 向量数据和栅格数据的转换

可以将纬度-经度向量数据转换为任意经度的网格数据，然后可以使用该网格数据制作栅格图。地图工具箱提供了进行此类转换的图形用户界面，也可以从命令行中进行转换，向量数据网格化的主要函数是 `vec2mtx`。如果向量数据由多边形组成，则网格化轮廓都是空的，可以用 `encodem` 函数离散它们。

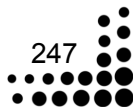
14. 用 GUI 光栅化多边形

可以把用 `getseeds` GUI 获得的种子点数据作为参数传递给 `seedm` 函数，用它填充网格化后的多个多边形。

23.2 基于地图工具箱的船舶定位研究

23.2.1 地图工具箱用于导航基本方法

导航是使设备或仪器从一个位置到另一个位置的规划、记录和控制等运动。这个词





来自拉丁词根 *navis* (“ 船舶 ”) 和 *agere* (“ 移动或方向 ”) 。通常以地理纬度和经度形式记录的地理信息是导航中的核心。工具箱中包含跨越广阔的全球范围内的投影坐标是实用的专门导航功能工具。

在陆地、水中和空中的导航涉及以下几部分任务：

- 利用已知的，固定的地标（飞行）获取位置。
- 通过恒星、太阳和月亮（天文导航）获取位置。
- 利用惯性导航、无线电导航或卫星导航等获取位置。
- 由过去已经的信息获取位置。

另外一个导航的任务涉及航海或飞行的计划，其中包括确定一个有效的途径（通常是大圈近似），气象回避（最优跟踪路由），并设置了预定的运动计划（轨道制定）。测绘工具箱支持这些导航活动。

可以通过下列导航工具箱函数进行角度和距离计算：

- `dreckon`。
- `gcwaypts`。
- `legs`。
- `navfix`。

为了使这些功能易于使用，并符合共同的航行实践，使用下列特定的约定：

- 以固定的角度运动。
- 以固定的相对距离运动。
- 以固定的速度运动。

导航追踪格式要求一个列向量，向量中变量是轨迹点的经度和纬度。航点是一个点，轨道通过该点，通常对应路径（或速度）的变化。航行轨道是由线段连接这些航点形成的，每一个线段称为航段。在这种格式中， n 个航段用 $n+1$ 个航点来描述，因为必须定义最后的端点，如图 23-5 所示。地图工具箱导航功能总是假定角度单位为度。

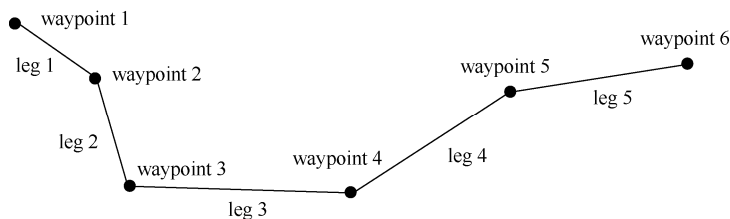


图 23-5 导航追踪格式

由图 23-5 可以看出，5 个航段需要 6 个航点。在导航轨道格式中，航点表示为两个 6×1 的向量，一个用于表示纬度，另一个表示经度。

导航的基本功能在于根据给定的信息来确定如何达到目的位置，并且能够避免中间的障碍。第一步需要做的是确定当前的位置。早期航海员主要通过观测海岛与船舶的相对关系来确定船舶的位置。通过视觉或雷达观测与岛屿之间相对距离从而确定船舶当前位置的方法被称为 *piloting*。通过经纬度或者与地标之间的相对距离能够完全确定物体的位置。

通过已知信息确定船舶当前位置需要进行计算，MATLAB 工具箱提供了 `navfix` 等函数来辅助计算。

在如何使用 `navfix` 函数之前，首先介绍根据已知信息确定当前位置的几种具体情况。

1. 已知三点并且知道与这三点的方位

这种情况如图 23-6 所示，图中 A、B、C 三点为已知的陆标，目前通过观测已知如下信息：

- A 点在船舶 150° 方位。
- B 点在船舶 180° 方位。
- C 点在船舶 90° 方位。

如果这三条线交于一点，则船舶的位置完全确定下来，但实际情况往往如图 23-6 所示，三条线交于三点，这同样是由于观测误差引起的。图中交于 1、2 和 3 点，这三点以及内部任意一点正确的可能性都接近。

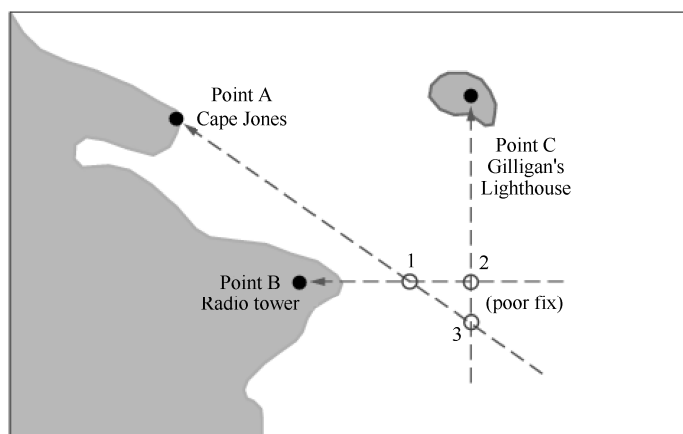


图 23-6 导航方式一

如果 1、2 和 3 点形成的三角形比较大，则说明观测误差较大，反之就较小。

由图可知，三条线相交形成三个交点，如果四条线相交则形成六个交点。交点越多信息越全，但计算量也越大，在误差接近的情况下，信息量多不一定对精度提高有帮助。

2. 已知三点并且知道与这三点之间的相对距离

这种情况如图 23-7 所示，图中 A、B、C 三点为已知的陆标，目前通过观测已知如下信息：

- A 点距离船舶 18 海里。
- B 点距离船舶 14 海里。
- C 点距离船舶 15 海里。

在这种情况下，分别绕 A、B 和 C 点做半径为 18、14 和 15 海里的圆，这三个圆交于一点，则船舶的位置可以完全确定下来。这是一种理想情况，实际有可能出现测量误差，这时三个圆不能交于一点，确定出来的船舶位置是一个区域。

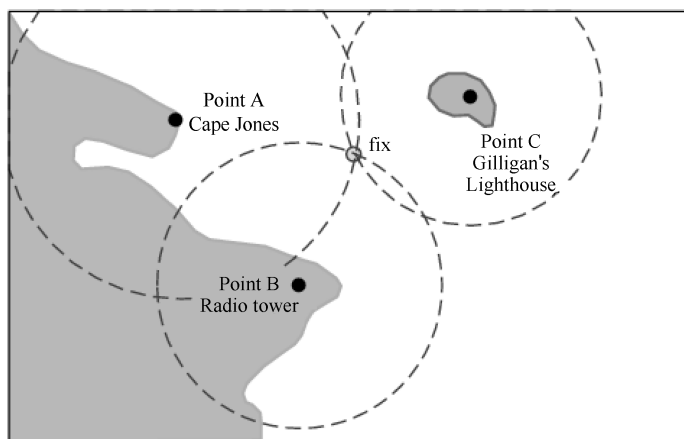


图 23-7 导航方式二

注：MATLAB 还提供了其他几种导航定位方法，感兴趣的读者可以参阅 MATLAB 帮助文档相关内容。

23.2.2 船舶最短路程规划实例

本例要求船舶按最短路径从美国弗吉尼亚州的 Norfolk (37°N , 76°W) 行驶到葡萄牙的 St. Vincent (37°N , 9°W)，且在目标点时利用 23.2.1 中介绍的定位方法进行定位。

大家都知道两个地理点之间的最短路径是一个大圆。但是在一般情况下，按照大圆路径，需要不断改变航向，这是不切实际的。另外一种简单的航行方式是沿着恒向线固定航向，但是一般这种方式较为费时。因此一般采用的方式是在航行中执行有限次的航向改变，而在前后两次航向改变之间采用恒向行驶方式。

从美国弗吉尼亚州的 Norfolk 到葡萄牙的 St. Vincent，是大西洋上最繁忙的路线之一。由于东恒向线轨道为 3213 海里，而最佳的大圆距离 3141 海里。

因此恒向线路径比大圆距离超过 2%，这是一个额外的 72 英里之旅。这个结果将造成 6 个小时的延迟。而在航运中，时间就是金钱，需要考虑新的航线方案。

这里考虑的是，用三段恒向线来近似大圆，此时行程的总距离 3147 海里。在这种情况下，只有一个半小时的延迟。下面是计算三种类型的航线 Norfolk 和 St. Vincent 对比的代码：

```
figure('color','w');
ha = axesm('mapproj','mercator',...
    'maplatlim',[25 55],'maplonlim',[-80 0]);
axis off, gridm on, framem on;
setm(ha,'MLineLocation',15,'PLineLocation',15);
mlabel on, plabel on;
load coast;
hg = geoshow(lat,long,'displaytype','line','color','b');
%定义 Norfolk 和 St. Vincent 的经纬度
```



```

norfolk = [37,-76];
stvincent = [37, -9];
geoshow(norfolk(1),norfolk(2),'DisplayType','point',...
    'markeredgecolor','k','markerfacecolor','k','marker','o')
geoshow(stvincent(1),stvincent(2),'DisplayType','point',...
    'markeredgecolor','k','markerfacecolor','k','marker','o')
% 计算并画出大圆上的一百个点
gcpts = track2('gc',norfolk(1),norfolk(2),...
    stvincent(1),stvincent(2));
geoshow(gcpts(:,1),gcpts(:,2),'DisplayType','line',...
    'color','red','linestyle','--')
% 计算并画出恒向线上的一百个点
rhpts = track2('rh',norfolk(1),norfolk(2),...
    stvincent(1),stvincent(2));
geoshow(rhpts(:,1),rhpts(:,2),'DisplayType','line',...
    'color',[.7 .1 0],'linestyle','-')
[latpts,lonpts] = gcwaypts(norfolk(1),norfolk(2),...
    stvincent(1),stvincent(2),3); % 计算三个航点
geoshow(latpts,lonpts,'DisplayType','line',...
    'color',[.4 .2 0],'linestyle','-')

```

由此产生的三条航线如图 23-8 所示。

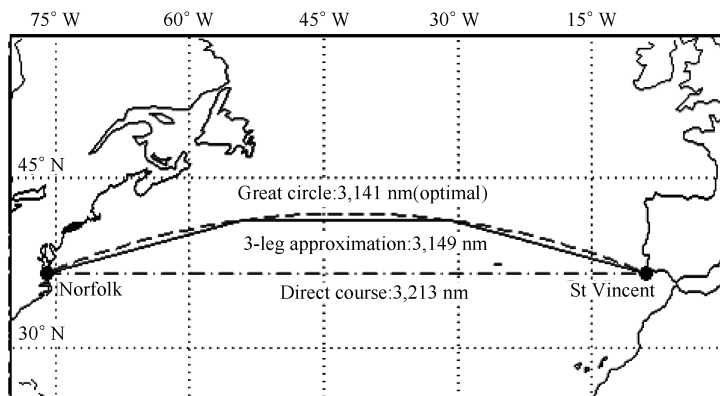


图 23-8 从 Norfolk 至 St. Vincent 三条航线对比

以上程序中函数 gcwaypts 使用的语法如下：

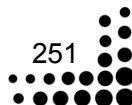
```
[latpts,lonpts] = gcwaypts(lat1,lon1,lat2,lon2,numlegs)
```

此功能的输入是起始和结束位置。numlegs 为恒向行驶的段数，这时默认为 10。输出列向量分别表示航向改变点经纬度。下面是上述程序中航向改变点的经纬度：

```

[latpts,lonpts] = gcwaypts(norfolk(1),norfolk(2),...
    stvincent(1),stvincent(2),3) % 计算三个航点
latpts =
    37.0000
    41.5076

```





```
41.5076
37.0000

lonpts =
-76.0000
-54.1777
-30.8223
-9.0000
```

驶近到 St. Vincent 时，需要对船舶的具体位置进行定位，下面给出采用 23.2.1 中介定位方法的程序：

```
% 用于定位的三点经纬度
lata = 34+3.1; lona = 47-56.2;
latb = 34+2.95; lonb = 47-55.9;
latc = 34+3.15; lonc = 47-55.95;

% 第一种导航定位方式
figure(2)
axesm('MapProjection','mercator','Frame','on',...
      'MapLatLimit',[36.85 37.25],'MapLonLimit',[-9.25 -8.85])
plotm([lata latb latc],[lona lonb lonc],...
      'LineStyle','none','Marker','pentagram',...
      'MarkerEdgeColor','b','MarkerFaceColor','b',...
      'MarkerSize',12)
[newlat,newlong] = navfix([lata latb latc],[lona lonb lonc],...
                          [289 135 26.5],[1 1 1])

plotm(newlat,newlong,'LineStyle','none',...
      'Marker','diamond','MarkerEdgeColor','r',...
      'MarkerFaceColor','r','MarkerSize',3)

% 第二种导航定位方法
figure(3)
axesm('MapProjection','mercator','Frame','on',...
      'MapLatLimit',[36.75 37.35],'MapLonLimit',[-9.35 -8.75])
plotm([lata latb latc],[lona lonb lonc],...
      'LineStyle','none','Marker','pentagram',...
      'MarkerEdgeColor','b','MarkerFaceColor','b',...
      'MarkerSize',12)
[newlat,newlong] = navfix([lata latb latc],[lona lonb lonc],...
                          [13 9 7.5],[0 0 0])
plotm(newlat(1:3,1),newlong(1:3,1),'LineStyle','none',...
      'Marker','diamond','MarkerEdgeColor','r',...
      'MarkerFaceColor','r','MarkerSize',3)
plotm(newlat(1:3,2),newlong(1:3,2),'LineStyle','none',...
      'Marker','diamond','MarkerEdgeColor','r',...
      'MarkerFaceColor','r','MarkerSize',3)
```

两种导航定位方法得到的定位结果分别如图 23-9 中 (a) 和 (b) 所示。图中三个蓝色星点表示用于定位的点，这些点一般是安装灯塔及无线发射装置的固定地理坐标。对于船舶而言，在定位前已经知道这三个点的经纬度。

图中红色点位根据已知点信息采用 23.2.1 中介绍定位方法得到的定位结果。从图中可以看出，在本例中，第一种方法定位的精度要高于第二种，但是第一种定位方法受到可见光限制。如果在雾天、黑夜等天气不好的情况下，此种方法应用即难于实现，而第二种方法具有更高的适应性。

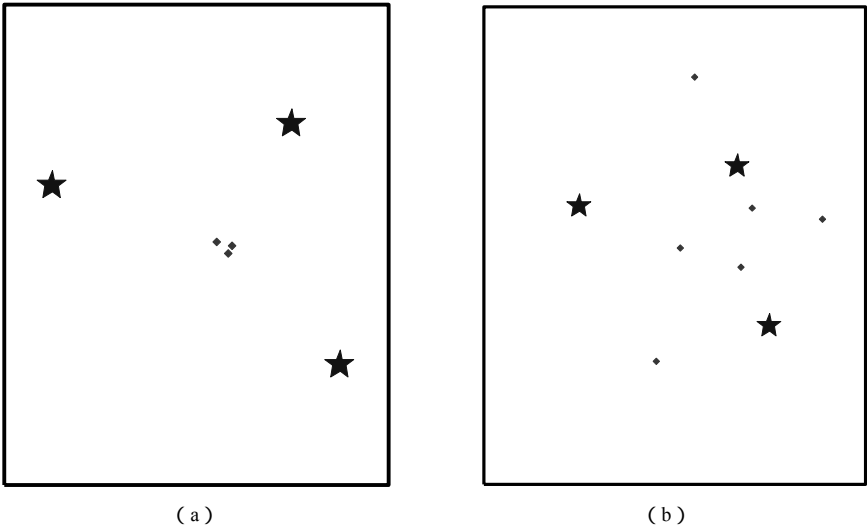


图 23-9 在 St. Vincent 两种导航定位的结果

23.3 本例小结

本例首先介绍了地图工具箱中创建地图和地理计算的基本知识，其次通过船舶定位实例介绍其图像处理工具箱在船舶最短路径规划和三点定位等方面的应用。



第 24 例 卫星星下点轨迹仿真

本例以卫星为对象，在介绍地图工具箱中地图投影和查看地图等基本知识的基础上，通过卫星星下点轨迹生成实例介绍其具体用法。通过本例学习，应掌握以下两点：

- 地图工具箱中地图投影和查看地图基本知识。
- 基于地图工具箱进行定位过程。

24.1 地图工具箱介绍（下）

24.1.1 地图投影

地球是一个球体，而不是平面，为了表示地球这样的二维曲面，必须在几何上将这个曲面变换为平面，这个变换就是地图投影。

1. 地图投影的定量属性

球体与多面体、锥体或圆柱不同，它不能展成平面，为了用二维平面描述球面，必须首先定义一个可展表明（即一个可以不通过拉伸或压缩就可以进行分割和展平的表明），并给出将球面的全部或部分对称表示到平面上的规则。任何方式的处理都不可避免导致这样或那样的变形。这种变化主要分为五个方面：形状、距离、方向、比例和面积。这五个方面定义如下。

1) 形状

当地图上任何点在任何方向上的比例尺相同时，形状保持局部不变，具备这种特点的投影称为是保角的。

2) 距离

从投影中心到地图上其他任何地方的距离保持不变的地图投影称为等距的。

3) 方向

在任何方向上方位角都能正确描述时，地图投影保持方向不变，许多方位投影都具有这个特点。

4) 比例

比例是图上两点之间的距离与地球表面对应两点之间实际距离的比率，没有一种方法能够在很大范围内保持一种比例不变，但可以做到将变化率维持在 1%~2%。

5) 面积

地图可以按比例描绘地球表面的区域，这样的地图投影方法称为等面积投影。

2. 几何表面

有三种标准类型的几何表面可用于地图投影：柱面、锥面和表面，此外还有三类组合形成的几何表面。

1) 柱面投影

柱面投影通过覆盖包围地球球体的圆柱来生成，它将地球图像首先投影到柱面上，然后从柱面展开到平面，如图 24-1 所示。展开时，纬线显示为水平线，经线显示为垂线。

```
landareas = shaperead('landareas.shp','UseGeoCoords',true);
axesm('balthsrt','Frame','on','Grid','on');
geoshow(landareas,'FaceColor',[1 1 .5],'EdgeColor',[.6 .6 .6]);
tissot;
```

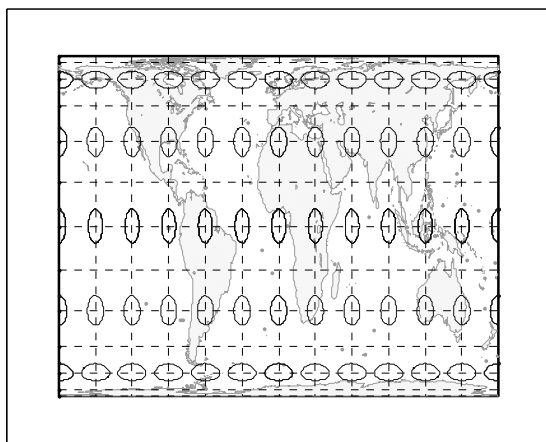


图 24-1 柱面投影图

除 balthsrt 外还有以下几种方式：

behrmann/ bsam/ braun/ cassini/ cassinistd/ ccylin/ eqacylin/ edqcylin/ giso/ gortho/ gstereo/ lambcylin/ mercator/ miller/ pcarree/ tranmerc/ trystan/ wetch。

2) 锥面投影

锥面投影是通过将地球表面投影到覆盖它的某个锥面上得到的。锥面的顶点位于地球极轴上，表面与某根纬线相切，圆锥将每个纬带作为单个锥面的一部分，锥面沿对应



的纬线与地球相切，如图 24-2 所示。

```
landareas = shaperead('landareas.shp','UseGeoCoords',true);  
axesm('eqaconic','Frame','on','Grid','on');  
geoshow(landareas,'FaceColor',[1 1 .5],'EdgeColor',[.6 .6 .6]);  
tissot;
```

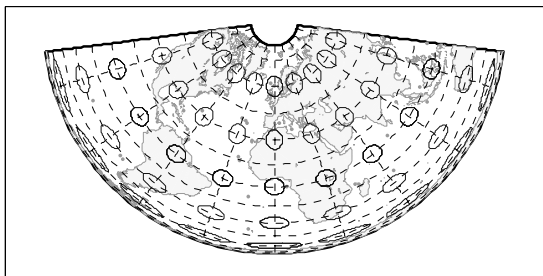


图 24-2 锥面投影图

3) 方位投影

方位投影将地球表面投影到平面上。对于极方位投影，平面与地球的某个极点相切，经线投影成放射状的直线段，纬线投影成圆心在极点的同心圆，如图 24-3 所示。

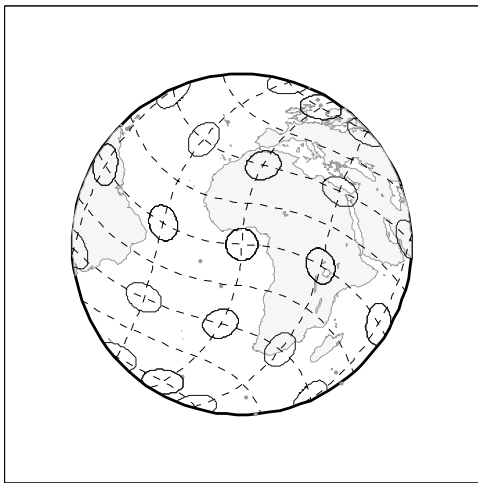


图 24-3 方位投影图

上面每种投影都可以作如下的改进。

(1) 除了与某根标准纬线相切外，柱面和锥面还可以切割两根纬线。同样，用于方位投影的平面也可以与地球相交而不仅仅是相切。

(2) 投影到几何表面上的中心点可以改变，可以选择从地心或沿极轴、从地球表面的对面或从空间中无限远点进行透视。

```
landareas = shaperead('landareas.shp','UseGeoCoords',true);
```




```
axesm('wiechel','Frame','on','Grid','on');
geoshow(landareas,'FaceColor',[1 1 .5],'EdgeColor',[.6 .6 .6]);
tissot;
```

24.1.2 创建和查看地图

地图工具箱提供了很多方法来控制地理空间数据的显示，下面介绍一些比较重要的函数和显示向量数据和栅格数据并与它们进行交互的相关界面。

1. 地图制作简介

使用地图工具箱显示地理信息可以像 MATLAB 中绘制表格或时间序列数据图形一样容易。除了接收地理/测量坐标数据外，大部分制图函数与 MATLAB 绘图函数近似。它们与 MATLAB 中具有相似功能的函数具有相同的名称，只是加上后缀“.m”，例如 plotm 函数就类似于 MATLAB 中绘图函数 plot。

地图工具箱管理当前地图中的大部分细节，它投影数据，将数据裁剪到合适的大小以适合指定的范围，并以不同比例显示最后生成的地图。使用工具箱，还可以添加一般图形具备的元素，如图框、网格线、坐标标注和文本标注等到当前地图中，如果改变投影属性，或改变投影类型，则地图工具箱用新设置重绘地图。

工具箱还使地图的修改和操作更加容易，可以从命令行或图形用户界面和属性编辑工具修改地图的显示特性和地图的对象，大部分地图显示函数都有图形用户界面。

1) 用 worldmap 和 usamap 函数显示地图

使用 worldmap 和 usamap 函数可以通过输入参数来控制地图的显示内容，下面给出两个例子：

(1) 创建南美的轮廓图，如图 24-4 所示。

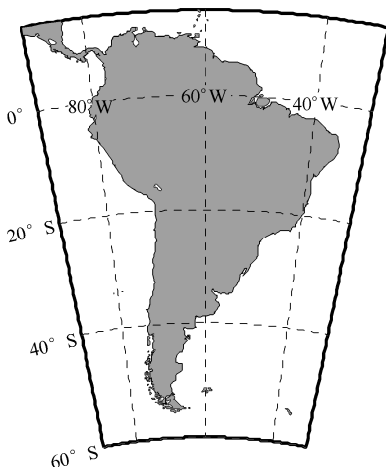


图 24-4 南美轮廓图



```
figure
worldmap('south america') % 创建地图
axis off
geoshow('landareas.shp', 'FaceColor', [0.5 0.7 0.5]) % 陆地形状数据
```

(2) 通过指定地理范围, 创建局部地图, 并设置文本标注, 如图 24-5 所示。

```
latlim=[37 40];
lonlim=[-78 -74];
usamap(latlim,lonlim)
geoshow('landareas.shp', 'FaceColor', [0.5 0.7 0.5])
textm(38.2,-76.1, '切萨皮克湾', 'fontweight', 'bold', 'Rotation',270)
```

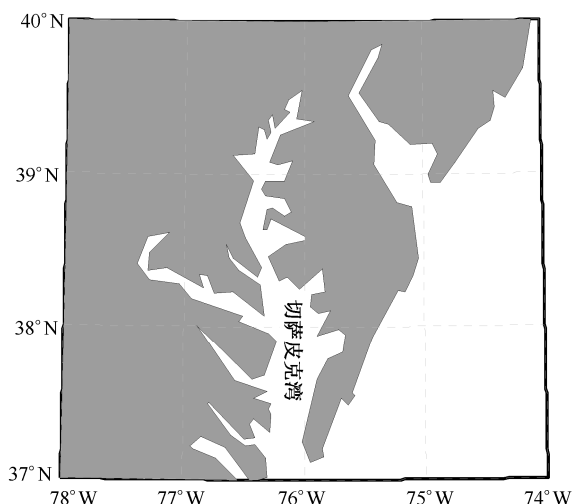


图 24-5 南美局部图

2) 坐标

可以用地图工具箱中的任何一个内部界面, 或者地图工具箱函数生成地图, 许多 MATLAB 图形都是用 axes 函数生成的, 例如:

```
axes
axes('PropertyName', PropertyValue,...)
h = axes(...)
地图工具箱提供了 axesm 函数, 它的功能与 axes 类似, 它的语法为:
axesm
axesm(handle)
axesm(PropertyName,PropertyValue,...)
axesm(ProjectionFile,PropertyName,PropertyValue,...)
```

调用没有变量的 axes 函数会弹出一个用户界面, 它列出了所有支持的投影类型并帮助定义它们的参数。可以用 axesmui 函数激活这个图形界面。

还可以用 maps 函数列出地图工具箱投影的名称、类型和 ID 字符串。axesm 函数创



建的坐标系与 MATLAB 中创建的坐标系具有相同的属性，另外还有与投影、比例和地理坐标位置有关的属性。

`axesm` 函数创建的地图坐标系将投影信息包含在一个可以通过它们的 `UserData` 属性获取的结构中。例如，键入下面的命令行可以查看所有相关属性。

```
h = axesm('MapProjection', 'mercator')
```

然后用 `getm` 函数提取出所有地图坐标系属性。

```
p = getm(h)
```

因为投影数据保存在坐标系结构的 `UserData` 字段中，也可以用一般的坐标系属性获得：

```
q = get(h, 'UserData')
```

用 `axesm` 函数创建的图形窗口包括与任何 MATLAB 图形窗口相同的工具和菜单集，默认是空的，即使工作空间中有地图数据时也是如此。对于某些属性，如网格、图框和坐标系标注等可以通过右键编辑功能进行打开和关闭操作。

可以定义多个独立的地图坐标系图形，但任何时候只有一个是激活的。就像使用 MATLAB 函数 `set` 和 `get` 可以获取和操作标准坐标系的属性一样，地图坐标系属性页可以用函数 `setm` 和 `getm` 进行获取和操作。

3) 创建一个不包含地图数据的地图坐标系（如图 24-6 所示）

```
axesm('Mapprojection','miller','Frame','on')
```

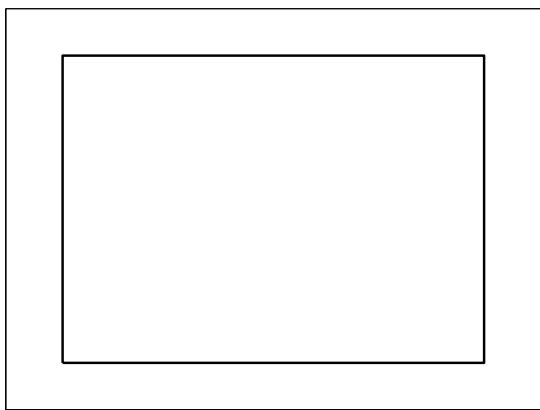


图 24-6 地图坐标系

4) 获取当前 `FineWidth` 属性

```
getm(gca,'FLineWidth')  
ans = 2
```



将直线的宽度设置为 4 磅，默认的单位是磅，可以设置为英尺、厘米或像素等其他单位，设置结果如图 24-7 所示。

```
setm(gca,'FLineWidth',4)
```

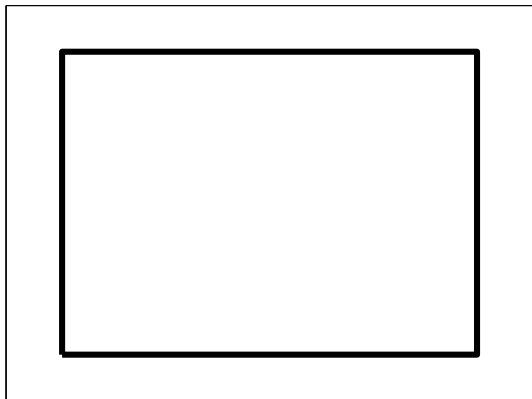


图 24-7 地图坐标系线宽值变化

5) 用 setm 函数同时设置任意属性

```
setm(gca,'FLineWidth',3,'MapProjection','eqdcylind')
```

此外，还可以用 getm 反映当前设置下所有地图坐标系属性，以及用 setm 函数显示属性集，包括其枚举值和默认值等。

2. 用地图制作工具箱函数显示向量数据

利用工具箱建立地图之后，还可以通过输入命令显示向量地理空间数据。下面介绍显示栅格地图数据的方法。

1) 把向量地图显示成直线对象

使用地图工具箱可以把向量地图数据显示成直线对象，相关函数见表 24-1。

表 24-1 把向量地图数据显示成直线对象的相关函数

函 数	功 能
contourm	绘地图数据的等值线图
contour3m	绘三维空间中地图数据的等值线图
linem	绘投影到地图坐标系中的直线对象
plotm	清除图形窗口中的内容并绘投影到地图坐标系的直线对象
plot3m	在三维空间中将直线对象投影到地图坐标系

2) 把向量地图显示成面片

向量地图数据可以显示为面片或填充多边形。patch 函数是地图制作工具箱中与 MATLAB 中的 patch 函数等价的函数。

表 24-2 列出了地图工具箱中可用的面片显示函数。

表 24-2 面片显示函数

函 数	功 能
fillm	填充二维地图多边形
fill3m	填充三维空间中的三维地图多边形
patchm	投影到地图坐标系的面片对象
patchesm	作为单独对象投影到地图坐标系的面片

24.2 卫星星下点轨迹图生成

1. 星下点轨迹公式

设卫星的位置变量为 x 、 y 和 z ，设经纬度分别为 lon 和 lat ，则它们之间的关系如式 (24-1) 所示：

$$\begin{cases} \text{lon} = \tan^{-1} \frac{y}{x} \\ \text{lat} = \tan^{-1} \frac{z}{\sqrt{x^2 + y^2}} \end{cases} \tag{24-1}$$

注意：当 x 为零时，根据 y 的正负可得 lon 为 90° 或 270° 。

2. 星下点轨迹生成程序

星下点轨迹生成程序如下所示。可以看出，它是一个 S 函数，将其与前面介绍的卫星轨迹生成程序结合在一起就可以显示星下点轨迹图，如图 24-8 和图 24-9 所示。

```
function [sys,x0,str,ts]=mdlInitializeSizes
sizes = simsizes;

sizes.NumContStates = 0;
sizes.NumDiscStates = 0;
sizes.NumOutputs = 0;
sizes.NumInputs = -1;
sizes.DirFeedthrough = 1;
sizes.NumSampleTimes = 1;    % at least one sample time is needed

sys = simsizes(sizes);
x0 = [];
str = [];
ts = [-1 0];
```



```
load coast
plot(long, lat); axis equal
set(gca, 'XLim', [-200 200], 'YLim', [-100 100])
grid on
hold on
sys = [];
```

```
function sys=mdlOutputs(t,x,u)

a=u(1);b=u(2);
plot(b*180/pi,a*180/pi,'r')

sys = [];
```

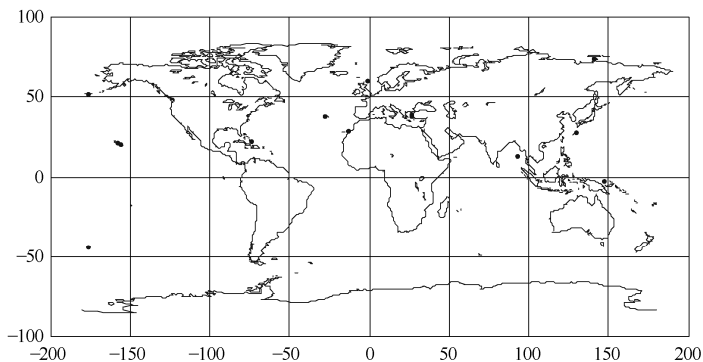


图 24-8 世界地图

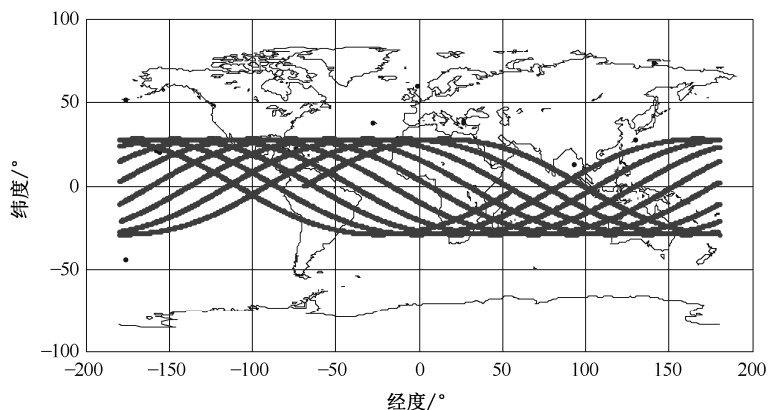


图 24-9 附加星下点轨迹的世界地图

24.3 本例小结

本例首先介绍了地图工具箱中地图投影和地图查看等的基本知识，其次通过卫星星下点轨迹实例介绍地图工具箱在绘制地图投影及地图查看等方面的应用。

第七部分 逻辑系统仿真实例



引言——逻辑系统简介

该部分为 MATLAB 工程仿真实例最后一部分,在这一部分将要介绍有限状态自动机 (finite-state machine, FSM) 及基于有限状态自动机的仿真实例。

有限状态机是指输出取决于过去输入部分和当前输入部分的时序逻辑程序。一般来说,除了输入部分和输出部分外,有限状态机还含有一组具有“记忆”功能的寄存器,这些寄存器的功能是记忆有限状态机的内部状态,它们常被称为状态寄存器。在有限状态机中,状态寄存器的下一个状态不仅与输入信号有关,而且还与该寄存器的当前状态有关,因此有限状态机又可以认为是组合逻辑和寄存器逻辑的一种组合。其中,寄存器逻辑的功能是存储有限状态机的内部状态;而组合逻辑又可以分为次态逻辑和输出逻辑两部分,次态逻辑的功能是确定有限状态机的下一个状态,输出逻辑的功能是确定有限状态机的输出。

有限状态机 (FSM) 或有限状态自动机或简称状态机,是表示有限个状态以及在这些状态之间的转移和动作等行为的数学模型。可以这样理解,系统的行为如果在不同的时间 (环境) 下,其工作不同,并且行为可以分成所谓的有限的状态以及不重叠的程序块时,系统显现出了状态行为。这其实说明了两件事情:一是离散的,二是有限的。

MATLAB 中实现有限状态机的工具箱是 Stateflow,它是 MATLAB 产品体系中非常重要的一个分支,它是在基于框图的动态系统建模仿真环境 Simulink 的基础上完成动态逻辑系统建模与仿真的可视化开发平台。Stateflow 能够对那些基于有限状态机理论的事件驱动系统进行建模与仿真,也能够对复杂逻辑系统进行建模与仿真。

在本书的第七部分,分四部分介绍 Stateflow 工具箱,其中第一部分为 Stateflow 编辑器中的状态模块与连接模块,在第 25 例中进行介绍,第二部分为 Stateflow 编辑器中的其他工具,在第 26 例中进行介绍,第三部分为基于 Stateflow 编辑器建立有限状态机对象的过程,在第 27 例中进行介绍,第四部分为 Stateflow 工具箱与 Simulink 仿真之间联系的基本原理,在第 28 例中进行介绍。在 29 例和 30 例中,再通过仿真进一步介绍基于 Stateflow 工具箱的仿真。



第 25 例 发射终止系统仿真

在发射航天器或火箭时，为安全起见，需要准备发射终止系统并实施一定的安全控制逻辑，这部分逻辑可以通过 Stateflow 实现。



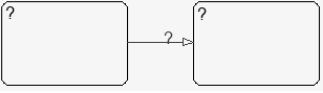
本例针对发射终止系统进行设计与仿真，通过本例应了解和掌握以下几部分内容：

- 了解 Stateflow 工具箱中状态模块基本概念和基本用法。
- 了解 Stateflow 工具箱中连接模块基本概念和基本用法。
- 了解基于 Stateflow 工具箱实现发射终止系统过程。

25.1 Stateflow 状态模块与连接模块简介

状态模块与连接模块的图标如表 25-1 所示。

表 25-1 状态模块与连接模块图标

名 称	工具栏图标	编辑窗口图标	功能描述
状态模块			用于表征系统的一个模式
连接模块	(无)		用于连接系统不同的状态模块

下面分别针对这两个模块进行介绍。

25.1.1 状态模块

每个有限状态机中都有若干个模式，而状态模块就是用于描述这些模式。

状态模块可以是活动的或非活动的。当模块处于活动状态时，Stateflow 建立的有限状态机即采用该模块对应的模式。通过事件及相应的条件可改变状态是否活动，在有限状态机运行的任何时候，都是由活动模块与非活动模块组合起来的。

状态有三个属性：从属关系、状态分类和状态标签。

1. 状态模块的从属关系

将一个状态模块置于另一个状态模块的内部，则前一个状态模块为后一个状态模块的子状态，而后一个状态模块为前一个状态模块的父状态。如果两个状态模块拥有一个共同的父状态模块，则这两个状态模块属于并列的状态模块。

图 25-1 所示为状态模块从属关系的示例，图中 Car_done 为 Car_made 和 Car_shipped 的父状态，Car_made 为 Parts_assembled 和 Painted 的父状态。同样的，Parts_assembled 和 Painted 为 Car_made 的子状态，Car_made 和 Car_shipped 为 Car_done 的子状态。

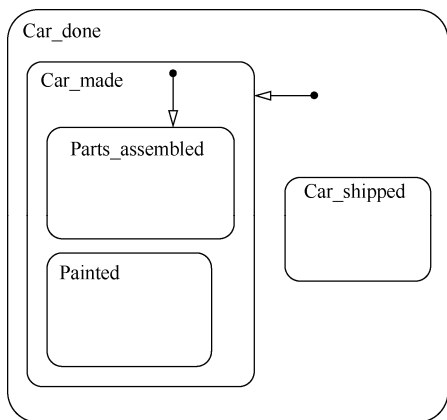


图 25-1 状态模块从属关系示例

此外，还可以用文字描述状态模块之间的从属关系，上图对应文字描述的关系如下所示：

```
/Car_done  
/Car_done. Car_made  
/Car_done. Car_shipped  
/Car_done. Car_made. Parts_assembled  
/Car_done. Car_made. Painted
```

上述文字描述的关系中，以“/”作为从属关系的开始，以“.”作为具有从属关系模块的连接符号。

2. 状态分类

状态机内部按照是否独占可分为独占模式、并行模式和混合模式。

1) 独占模式

处于独占模式时，在任一时刻处于并列位置的活动状态模块有且只有一个，如图 25-2 所示。

图 25-2 中初始默认（初始默认连接模块将在后面介绍）进入的状态为 B，在事件 E1 作用下进入状态 A，状态 A 默认进入的子状态为 A1，在事件 E2 作用下进入状态 A2。在任一时刻，A 和 B 不能同时处于活动状态，A1 和 A2 也不能同时处于活动状态。

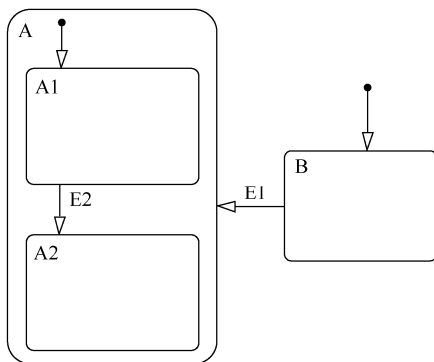


图 25-2 独占模式示例

2) 并行模式

处于并行模式时，可使两个或多个状态处于活动状态，此时状态模块边界用虚线表示，如图 25-3 所示。

图中当父状态 A 处于活动状态时，两个子状态 A1 和 A2 同时处于活动状态。

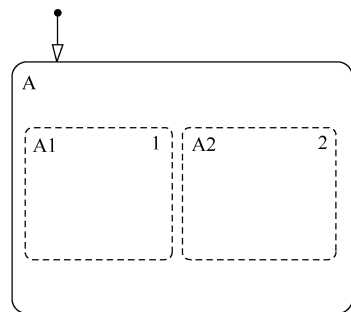


图 25-3 并行模式示例

3) 混合模式

状态机中既有并行模式，也有独占模式，如图 25-4 所示。

图中当 A 处于活动状态时，B 和 C 属于并行模式，可同时进入活动状态，而当 C 进入活动状态时，C1 和 C2 属于独占模式，只能同时有一个进入活动状态。

3. 状态标签

图 25-5 所示为状态标签示例。

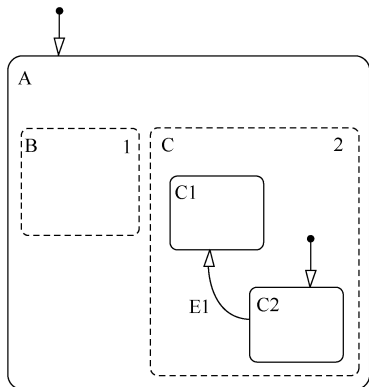


图 25-4 混合模式示例

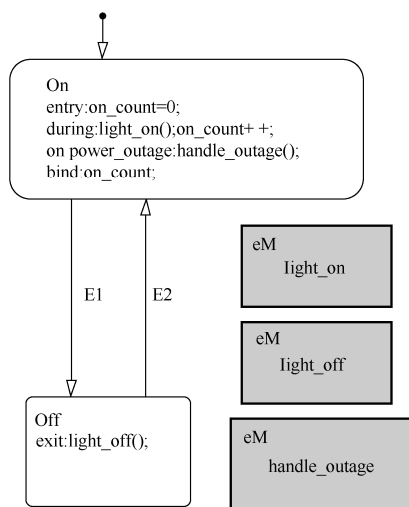


图 25-5 状态标签示例



由图 25-5 可以看出状态标签位于状态模块左上角，采用的语法格式如下：

```
name/  
entry: entry actions  
during: during actions  
exit: exit actions  
bind: events, data  
on event-name: on event_name actions
```

状态标签中关键词共六个：name、entry、during、exit、bind 和 On event_name。下面结合示例分别予以介绍。

1) name

状态标签始于状态名称，状态名称可选用“/”结束。图中两个状态的名称分别为“On”和“Off”。状态模块的从属关系为状态名称起到了简化作用，这是因为状态名称的范围可以设置为父状态包含部分。

2) entry

entry 可用 en 代替，该关键词表示在进入状态时执行的命令。在图中，当 On 模块进入活动状态时，on_count 被设置为 0。

3) during

during 可用 du 代替，该关键词表示模块处于活动状态时执行的命令。在图中，当 On 模块处于活动状态时，执行 light_on()和 on_count++。

4) exit

exit 可用 ex 代替，该关键词表示模块退出活动状态时执行的命令，在图中没有该关键词对应的命令。

5) bind

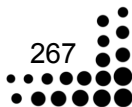
bind 用于表示将变量绑定于某状态，在上例中，on_count 被绑定于状态模块 On，此时 on_count 只有在状态模块 On 及其子状态中可见，而状态模块 Off 无法看到 on_count。

6) On event_name

该关键词使在状态活动时产生事件，在上例中，当状态模块 On 处于活动状态时，handle_outage 被执行。

25.1.2 转移

转移是一条有一个箭头的线，它链接一个图形对象到另一个对象。在大多数情况下，转移代表系统通过从一个模式（状态）的对象到另一个对象。转移通常连接源和目标对



象。源对象是过渡的开始位置，而目标对象是转移结束位置。如图 25-6 所示的转移示例，显示了一个从源状态 B，过渡到目标状态 A。

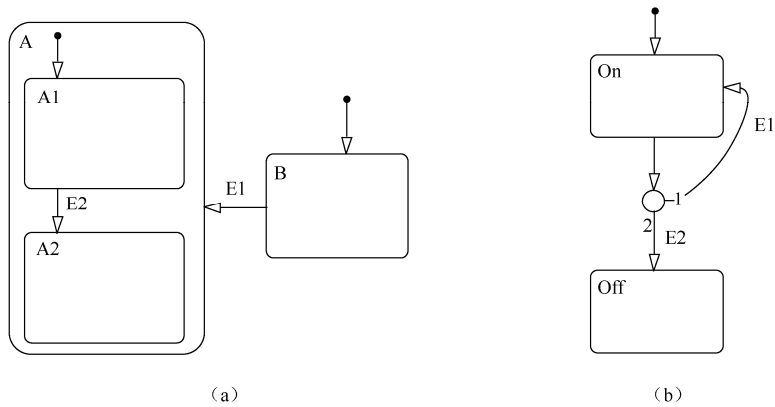


图 25-6 转移示例

转移有三个基本属性：从属性、标签和有效性，下面分别针对这三方面内容予以介绍。

1. 从属性

转移的从属关系通过转移父状态模块、转移源和转移目标进行描述。转移父状态模块包含转移源和转移目标最低一层父结构，如图 25-7 所示，图中转移从属关系可以通过表 25-2 进行描述。

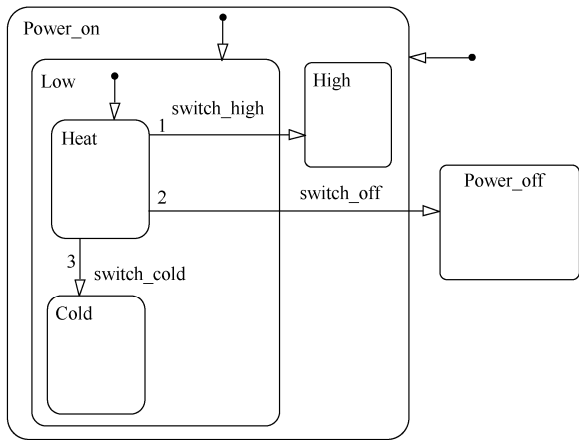


图 25-7 转移的从属特性

表 25-2 转移从属关系

转 移 标 签	转移父状态模块	转 移 源	转 移 目 标
switch_off	/	/Power_on.Heat	/Power_off
switch_high	/Power_on	/Power_on.Heat	/Power_on.High
switch_cold	/Power_on.Low	/Power_on.Heat	/Power_on.Low.Cold

2. 转移标签

转移是 Stateflow 编辑器中最常见的图形元素之一，转移描述的是有限状态系统内的逻辑流。当转移发生时，源状态变为非活动的状态，目标状态变为活动的状态。只有当源模块为激活状态时，状态转移才是有效的，无效的状态不会被执行。状态转移的执行属性也是由其标识确定，标识显示在曲线附近，默认标识为“？”（在非编辑状态下不显示），表示在任何事件发生时都将执行该转移。标识的一般格式为：

```
event[condition]{condition_action}/transition_action
```

其中事件（event）是 Stateflow 非图形对象的一种。在有限状态机中，只有在事件发生时，才可能去执行相应转移，因此 Stateflow 的模型又叫做事件驱动系统。位于“[]”中的内容是条件，用于转移决策的逻辑判断。只有在相应事件发生且条件也满足时，相应的转移才可能执行。紧接在条件后面“{}”中的内容就是条件动作，条件动作是在条件满足时就立即执行的某些表达式，例如赋值运算等。转移动作（transition_action）是整个转移标签的最后部分，位于“/”后面的内容都是转移动作。转移动作只有在整个转移通路都有效时才能够执行。

转移语句示例如图 25-8 所示。

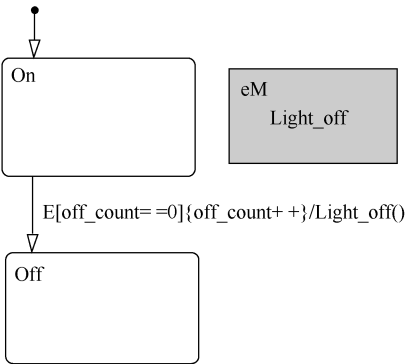


图 25-8 转移语句示例

3. 转移的有效性

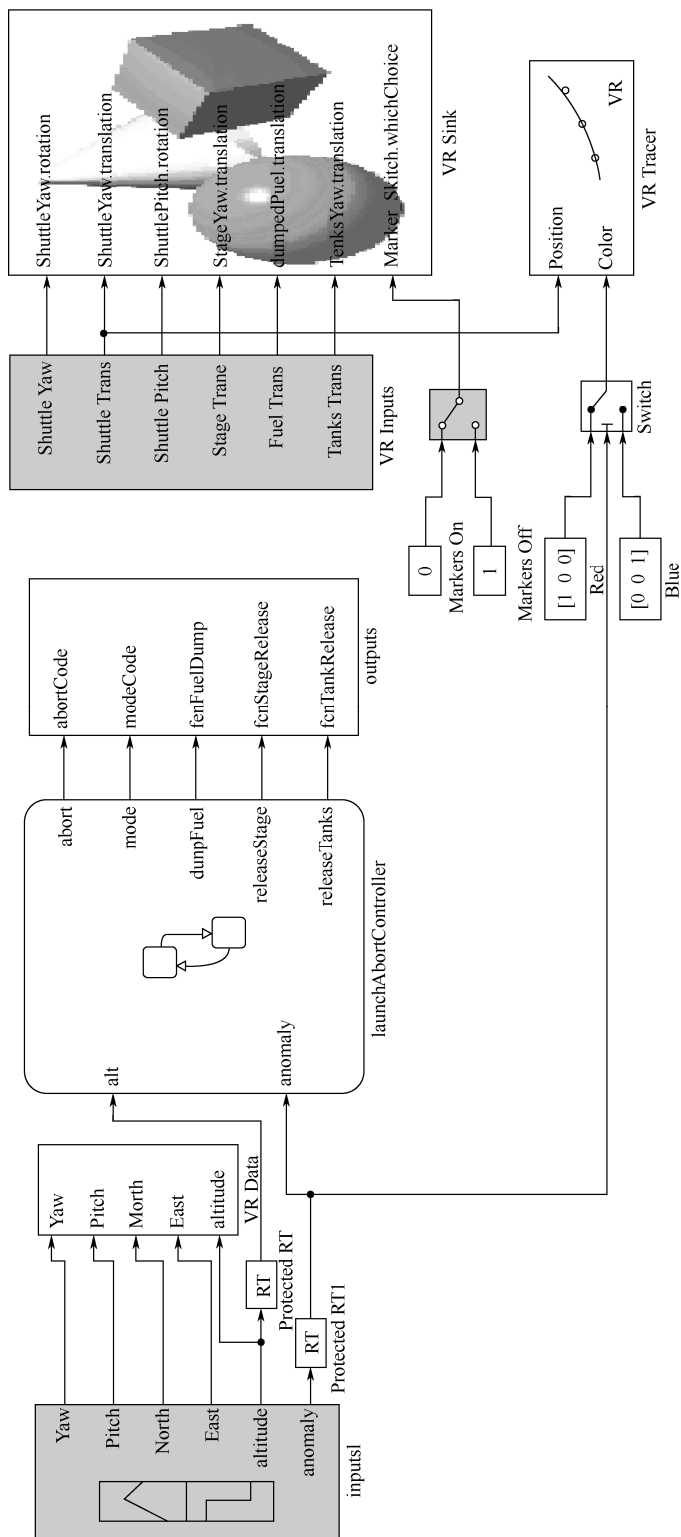
在大多数情况下，转移是有效的，需要源状态处于活动状态以及转移标签是有效的。默认转换是不同的，这是因为它没有源状态。表 25-3 列出了有效的转移标签可能的组合。

表 25-3 有效转移标签可能的组合

转移标签	有效情况分类
只有事件	对应事件发生
事件和条件	事件和条件都为真
只有条件	任意事件发生且条件为真
处于活动状态	任意事件发生
无特殊指定	任意事件发生

25.2 发射终止系统

发射终止系统用于控制在发射过程中如果出现异常状况将发射终止的逻辑转换，系统如图 25-9 所示。



Copyright 2007-2009 The MathWorks, Inc.

图 25-9 发射终止系统

发射终止系统由信号输出部分、逻辑转换部分和输出显示部分等组成，其中逻辑系统由 Stateflow 建立，其内部结构如图 25-10 所示。

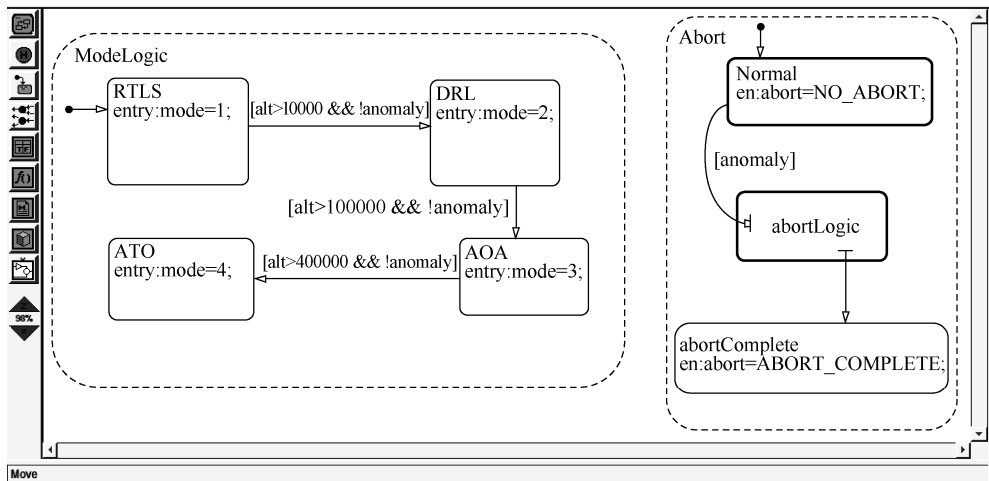


图 25-10 发射终止逻辑转换系统

由图 25-10 可以看出，发射终止逻辑转换系统由两个并列的状态模块 ModeLogic 和 Abort 组成。

其中状态模块 ModeLogic 表示没有异常状况下系统的逻辑转换，系统按照高度（alt）由模式 1，经模式 2 和模式 3，转换至模式 4。

而状态模式 Abort 表示系统在存在异常状况时，系统的逻辑转换过程，逻辑转换分为三个步骤：Normal、abortLogic 和 abortComplete。其中 abortLogic 内部的逻辑结构如图 25-11 所示，由图可以看出终止逻辑分为三个步骤，按照顺序分别执行 FuelDump、TankRelease 和 StageRelease。

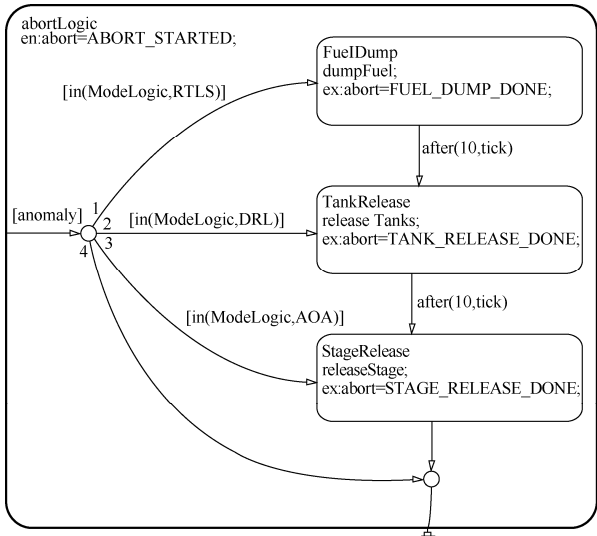


图 25-11 终止逻辑



25.3 本例小结

本例学习了 Stateflow 中状态模块和转移模块的基本概念，并通过发射终止系统仿真使对这两个模块加深了解，为继续深入学习 Stateflow 奠定基础。

第 26 例 月球登陆器自动驾驶仪仿真

安装在“阿波罗”指挥舱和登月舱内的数字计算机是麻省理工学院仪表实验室设计的，设计的目的是用来进行制导和导航。在飞行主动段，飞行器的俯仰和偏航控制通过指挥舱的推力供给系统的常平架固定式定向发动机偏转来实现。绕横滚轴的姿态控制则用控制系统的喷气发动机完成。在对指挥姿态和所测姿态之间的计算误差的响应中，常平架伺服指令的计算就是推力矢量控制数字自动驾驶仪的功能，这种功能通过指挥舱计算机来实现。








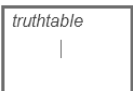
本例在介绍基于 Stateflow 月球登陆器自动驾驶仪的基础上，进一步介绍 Stateflow 工具箱其他模块的概念和基本用法，通过本例学习，需掌握以下几点内容：

- 了解 Stateflow 其他模块的概念和基本用法。
- 加深对转移的理解，了解其部分高级应用。
- 了解基于 Stateflow 实现的月球登陆器自动驾驶仪功能。

26.1 Stateflow 其他模块的概念和基本用法


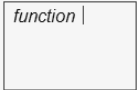



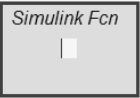
首先给出 Stateflow 其他模块的图标，如表 26-1 所示。

表 26-1 Stateflow 其他模块的图标

名 称	工具栏图标	编辑窗口图标	功 能 描 述
历史记录模块			用于记录历史状态信息
默认转移模块			用于设置存在歧义情况下默认进入的模块
转移连接点			用于连接转移线
表格函数			利用表格的形式定义函数



续表

名 称	工具栏图标	编辑窗口图标	功 能 描 述
函数			Stateflow 函数
盒子模块			包含其他模块的函数
MATLAB 函数			用 MATLAB 语言定义函数

下面对其中的内容分别予以介绍。

26.1.1 默认转移模块

默认转移模块用于在有歧义时，指定默认进入的状态模块。默认的转移有一个目标，但没有源对象。

在画默认转移时，首先点击工具栏中默认转移工具，然后在需要默认转移的状态模块内部单击，使默认转移位于状态模块内部，然后拖动鼠标使默认转移图标箭头落在目标状态模块或节点上。

注意：在 Stateflow 编程时，容易出现的一个错误是，建立了多个独占模块而忘记设置默认转移，从而使系统产生歧义。

下面给出默认转移的三个例子，如图 26-1 至图 26-3 所示，其中例 1 演示指向状态模块的默认转移，例 2 演示指向节点的默认转移，例 3 演示默认转移的标签。

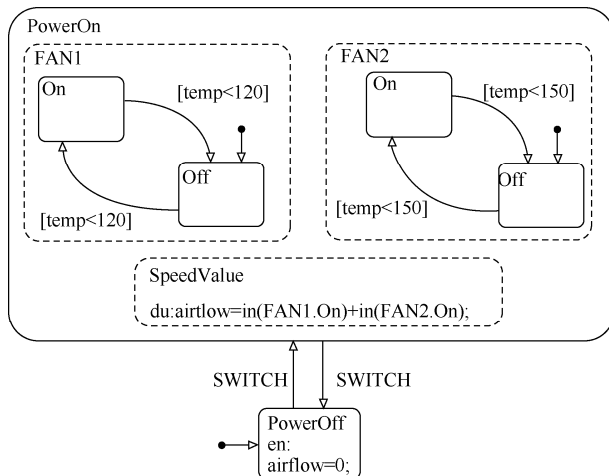


图 26-1 默认转移例 1

图 26-1 给出默认转移例 1，图中存在三个默认转移：第一个默认转移指向模块 PowerOff，第二个默认转移指向模块 FAN1，第三个默认转移指向模块 FAN2。如果没有指向 PowerOff 的默认转移，则当 Stateflow chart 被激活时，系统无法确认是先执行 PowerOff，还是先执行 PowerOn。

图 26-2 给出默认转移例 2，图中默认转移指向节点，再通过转移条件来选择执行模块 P 或模块 N。

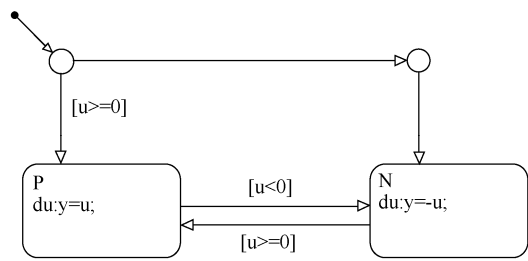


图 26-2 默认转移例 2

图 26-3 给出默认转移例 3，由例 3 可以看出默认转移也可以设置标签，例 3 中默认转移设置当执行时设置 P=10 且 V=15。

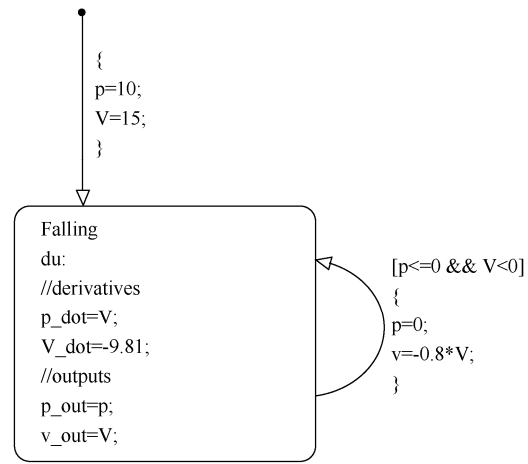


图 26-3 默认转移例 3

26.1.2 历史节点

历史节点用于记录仿真上一时刻的状态并辅助下一时刻的决定，如图 26-4 所示。

图中 Power_On 有一个历史节点，包含两个子状态 Low 和 High。该系统可以进入 Power_on.Low 或 Power_on.High。首次进入 Power_On 时，因为 Power_on.Low 有默认转移，因此该模块首先被执行。在一点之后，如果状态 Power_on.High 是活动状态且事件 Switch_off 发生，Power_On 退出并进入状态 Power_Off。然后发生事件 Switch_on，因为



Power_on.High 是上一个活动状态，因此进入 Power_On 后首先进入状态 Power_on.High。

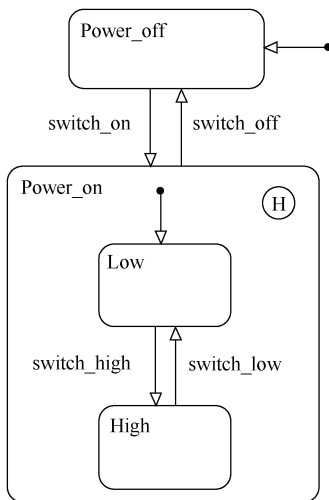


图 26-4 历史节点例子

26.1.3 连接节点

连接节点是系统中转换通路的判断点和汇合点，它可以简化 Stateflow 图表表示并加速代码生成。与节点相连的称为转换段。一旦转换段完成从一个状态到另一个状态的转换，转换段会累积构成一段完整转换。通常连接节点用来构成图形化的分支程序，如 if-then-else 结构等。

图 26-5 所示为连接节点例 1。例 1 演示利用连接节点实现 if-else-then 结构语法。在图中，当小于 2s 时，进入模块 Fast，当大于 2s 小于 5s 时，进入模块 Good，当大于 5s 时，进入模块 Slow。

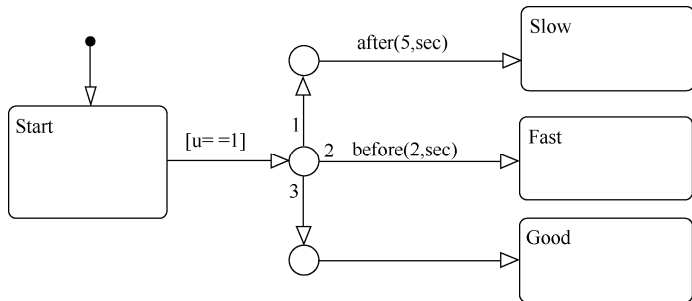


图 26-5 连接节点例 1

图 26-6 所示为连接节点例 2。例 2 演示利用连接节点实现 for 循环语法结构。在图中，当 E_one 事件发生时，首先进行循环，循环时执行函数 func1()，执行 10 次后进入状态模块 B。

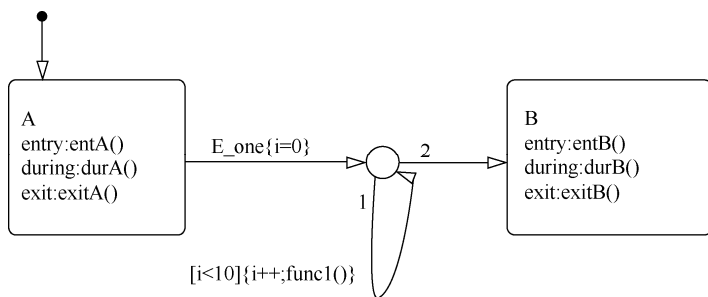


图 26-6 连接节点例 2

图 26-7 所示为连接节点例 3。例 3 演示综合利用连接节点实现 if-else-then 和 for 循环功能。

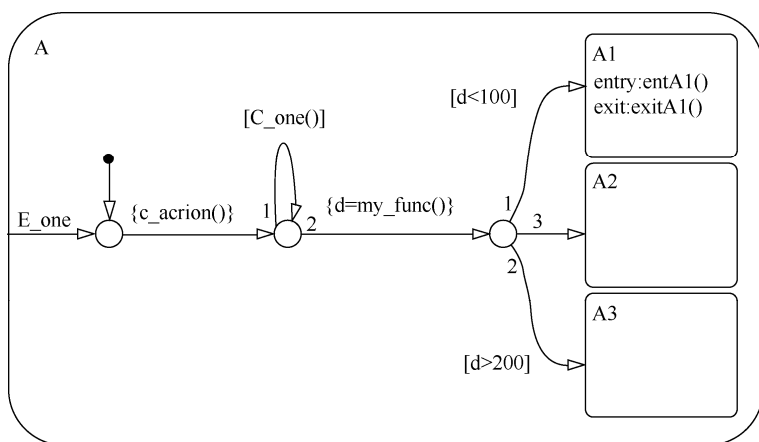


图 26-7 连接节点例 3

图 26-8 所示为连接节点例 4，该例演示利用节点实现一个源状态指向多个目标状态。

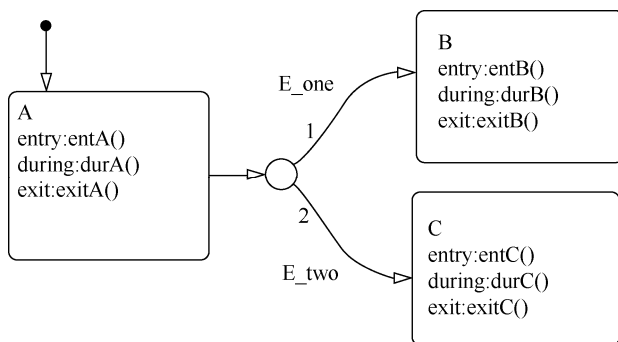


图 26-8 连接节点例 4

图 26-9 所示为连接节点例 5，该例演示多个源状态指向同一个目标状态。

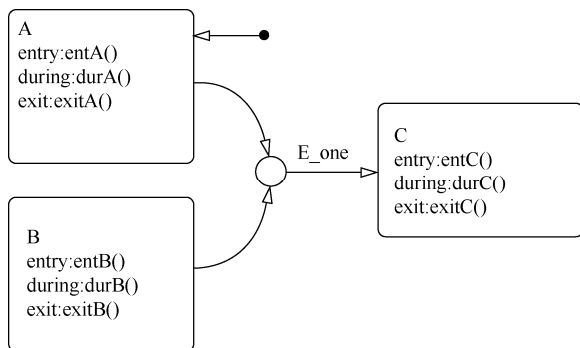


图 26-9 连接节点例 5

26.1.4 盒子模块

盒子模块包含多个状态模块和连接模块等，如图 26-10 所示。

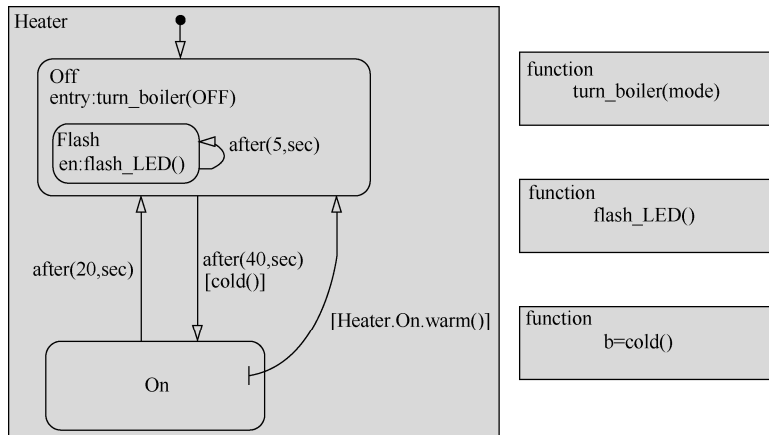


图 26-10 盒子模块例子

图中盒子模块 Heater 包含两个状态模块 Off 和 On 以及它们之间的转移。

26.1.5 连接分类

上面介绍了连接的基本概念，这里对连接进一步进行介绍，主要对连接进行分类。图 26-11 至图 26-15 所示为连接的五个分类示例。

图 26-11 演示了状态模块之间的连接。

图 26-12 演示了状态模块与连接节点之间的连接。

图 26-13 中的转移用于连接不同级别的状态模块。

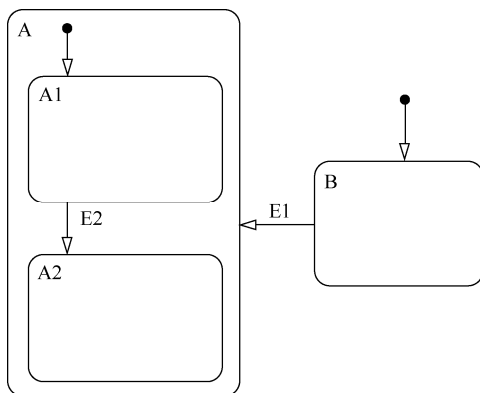


图 26-11 状态模块之间连接

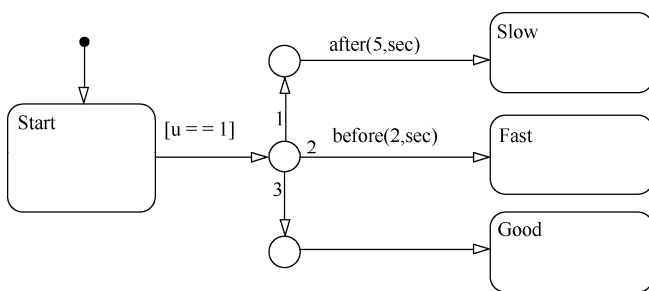


图 26-12 状态模块和连接节点之间连接

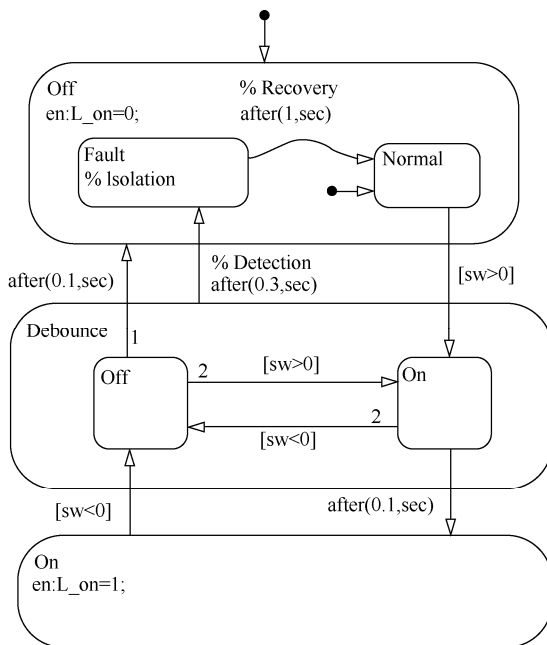


图 26-13 不同级别状态模块转移

图 26-14 中共有四个转移，都是由状态模块 TotalDynamics 转移到自身。

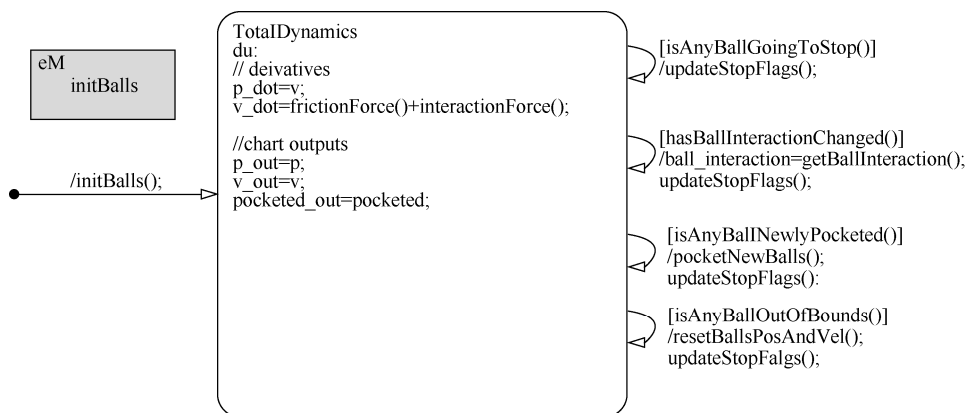


图 26-14 返回到自身的转移

图 26-15 中转移发生在状态模块 A 内部，这类转移有利于简化内部逻辑。

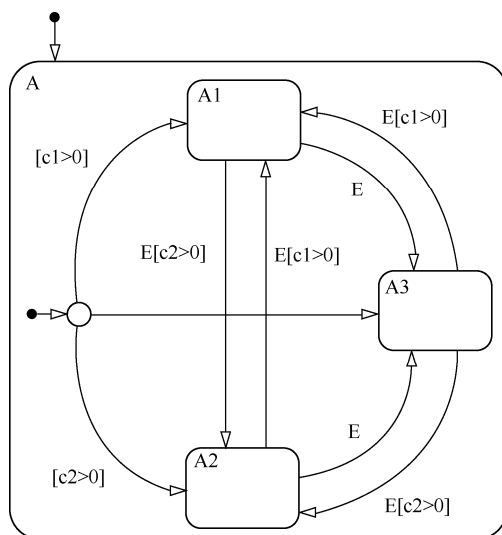


图 26-15 状态模块内部转移

26.2 月球登陆器自动驾驶仪仿真

月球登陆器自动驾驶仪仿真系统如图 26-16 所示。

月球登陆器自动驾驶仪仿真系统主要涉及三方面内容：数据存储共享系统、动力学系统和开关逻辑生成系统，下面对这三方面内容分别予以介绍。

?

[illegible]

26.2.1 数据存储共享系统

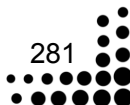
The diagram illustrates the data flow of the Phase Plane Plot (PP) module. It begins with 'Read data' entering a box labeled 'e_edot'. This box points to a 'Phase Plane Plot' box, which contains 'Pos. Emer'. From 'Pos. Emer', the flow goes to a 'Two Jet' selection stage. This stage has two outputs: 'Four Jet' and 'Two Jet or Four Jet Yaw Couples'. The 'Two Jet or Four Jet Yaw Couples' output points to a 'Write Var' box. The 'Write Var' box points to a 'Nominal Two Jet Couples' box. The 'Nominal Two Jet Couples' box points to a 'PitchRollJets' box. The 'PitchRollJets' box points to an 'Initialize Data Stores' box. The 'Initialize Data Stores' box branches into two outputs: 'NorJets' and 'PitchRollJets'.

图 26-17 数据共享系统

26.2.2 动力学系统

月球登陆器动力学模型如图 26-18 所示, 其采用的公式如式 (26-1) 所示:

$$I\dot{\varphi} + \varphi \times I\varphi = T \quad (26-1)$$



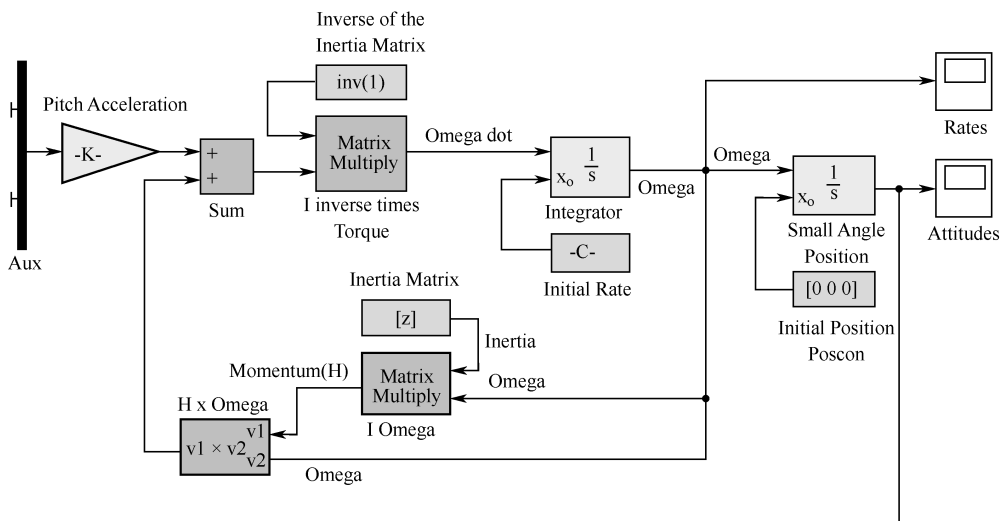


图 26-18 月球登陆器动力学系统

26.2.3 开关逻辑生成系统

开关逻辑生成系统如图 26-19 和图 26-20 所示。其中图 26-19 包含滤波系统及指令生成系统，图 26-20 为基于 Stateflow 实现的具体开关逻辑。

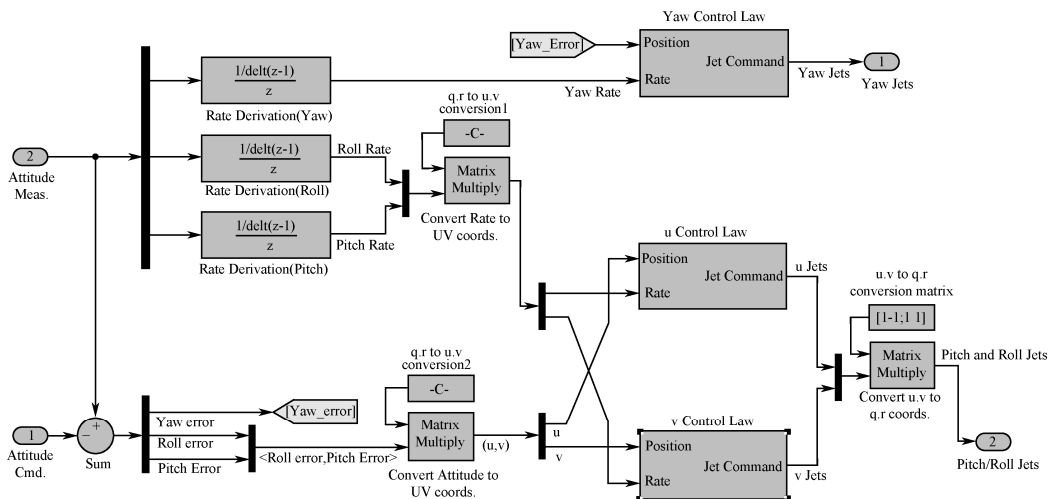


图 26-19 滤波及指令生成系统

图 26-20 中有四个主要的状态模块和另外四个辅助的状态模块。

四个主要的状态模块为 Fire_region_1、Coast_region_1、Fire_region_2 和 Coast_region_2，这四个状态模块分别表示两条开线和两条关线。当达到开线时，推力器打开阀门开始喷气控制，当到达关线时，推力器关闭阀门停止喷气控制。开线关线的具体位置设置需根据实际情况来确定。

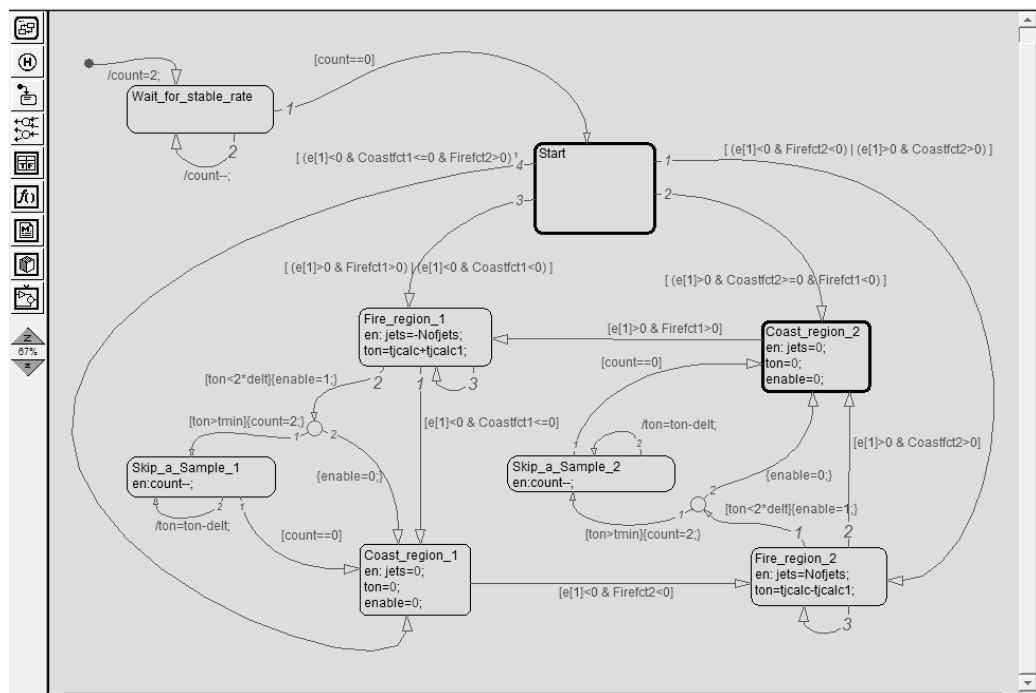


图 26-20 开关逻辑

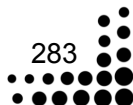
四个辅助的状态模块为 Wait_for_state_rate、Start、Skip_a_sample_1 和 Skip_a_sample_2。其中 Wait_for_state_rate 模块用于判断是否进入开关逻辑生成阶段，Start 模块用于判断开始开关逻辑生成阶段后，首先进入哪个主要的状态模块，Skip_a_sample_1 和 Skip_a_sample_2 用于分析采样时间是否大于执行时间。

本例所述的指挥舱和自动驾驶仪已用于“阿波罗”飞船中，飞行结果表明，这些自动驾驶仪的预想结果和测量结果（包括姿态误差动态特性、发动机偏转和法向速度误差）相当一致。

在载人飞船内第一次采用数字自动驾驶仪结果表明，数字计算机就完成自动驾驶仪任务而言，在许多方面都比模拟系统好，计算机不仅提供所需动态特性，而且能够完成模拟系统不容易完成的任务，这些任务是在推力系统发动机工作头几秒内自动估算和引入发动机平衡校正，在每次发动机工作结束时自动存储发动机平衡估算值，以便在下次发动机工作时使用，自动改变自动驾驶仪增益，以补偿推进剂消耗影响，在登月舱自动驾驶仪进行复杂的模型变换，以及飞行中，利用更换可清除存储器内系数的方法改变登月舱高带宽滤波器，模拟系统也能完成上述各项任务，但需要有更为复杂的定时、逻辑、取样、保持、参数变换等工作电路。

26.3 本例小结

本例在学习 Stateflow 其他模块的基本概念和用法的基础上，实现了月球登陆器自动驾驶仪仿真，通过本例学习对 Stateflow 基本模块有较为全面的了解，下面可以继续学习 Stateflow 的一些其他应用。





第 27 例 飞机俯仰轴容错控制仿真

飞机俯仰轴控制通过升降舵实现，为保证可靠性，升降舵一般做冗余备份，在部分升降舵发生故障时，需要改变控制逻辑以实现稳定控制，这一检测出故障升降舵并重新制订分配方案的逻辑可以通过 Stateflow 实现。

本例在介绍如何基于 Stateflow 工具箱建立有限状态机的基础上，以飞机为对象，介绍建立俯仰轴容错控制逻辑并仿真的过程。通过本例学习，需要掌握以下两点：

- 基于 Stateflow 建立有限状态机的过程。
- 基于 Stateflow 的飞机俯仰轴容错控制系统仿真。

27.1 基于 Stateflow 建立有限状态机过程

建立基于 Stateflow 的仿真模型分为两个步骤。

- (1) 建立 Stateflow Chart 内部结构。
- (2) 建立 Stateflow Chart 输入/输出变量。

下面分别针对这两部分内容予以介绍。

27.1.1 建立 Stateflow Chart 内部结构

建立过程分为以下几个步骤。

1. 创建 Stateflow Chart 模块

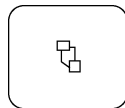
首先与打开其他工具箱类似，在 Simulink 中找到 Stateflow 并单击，可看到 Chart 模块，新建 mdl 文件并将 Chart 模块拖拽至 mdl 文件中，Chart 模块外观如图 27-1 所示。

双击打开 Chart 模块，如图 27-2 所示。由图可以看出这是一个包含工具箱的可视化图形建模界面。界面分为标题栏、菜单栏、工具栏、状态栏，作图区域，在创建对象之后，还可以右键选择对象来设置部分属性。

在图 27-2 中可以逐步建立需要的有限状态机模型。

2. 建立状态模块

建立状态模块分为以下几个步骤。



Chart

图 27-1 Chart 模块外观图

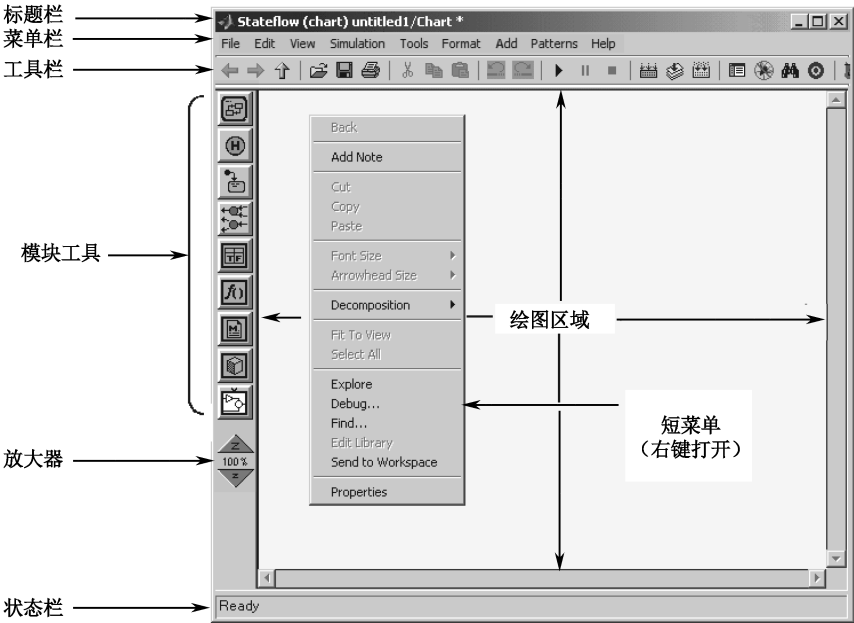


图 27-2 Stateflow Chart 框图

- (1) 单击状态模块按钮。
- (2) 将鼠标移至作图区域，选择合适的位置并单击和拖动鼠标完成状态模块建立，此时状态模块中只有左上角有一个“？”号。
- (3) 单击“？”号，可进行标签输入。

3. 建立转移线

- 建立转移线分为以下两个步骤。
- (1) 将鼠标移至一个状态模块边界线上，此时鼠标会变成圆形，单击鼠标左键并按住往外拖拽，则转移线产生，此时保持鼠标左键按住状态直到目标状态模块边界线，松开鼠标左键，则一条带单箭头的转移线生成。
 - (2) 单击转移线，此时转移线高亮并出现“？”号，单击“？”号，并输入标签。
- 右键转移线选择属性，出现对象框，如图 27-3 所示，对话框中出现的属性及其描述如表 27-1 所示。

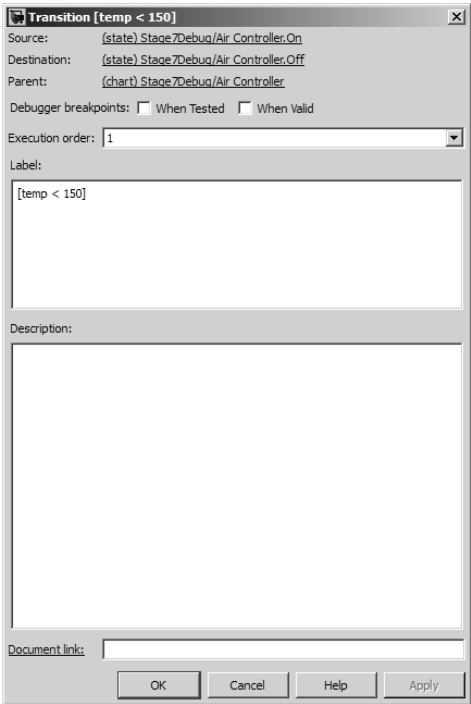


图 27-3 转移线属性



表 27-1 转移线属性

属 性	描 述
转移源	转移的源状态模块
转移目标	转移的目标状态模块
父状态模块	转移的父状态模块
调试中断点	当需要调试转移时，中断的地方
执行阶数	Chart 执行转移的阶数
标签	转移的标签
描述	对命令的文字描述
文档链接	网页位置

4. 建立默认转移线

建立默认转移线分为以下三个步骤。

(1) 单击默认转移按钮。

(2) 将鼠标移至作图界面，则出现一条一端带圆点另一端为箭头的默认转移线，移动鼠标直到箭头位于目标状态模块边界线上面，单击鼠标左键，则生成默认转移线。

(3) 单击默认转移线，则默认转移线高亮并出现“？”号，单击“？”号，并输入标签。

27.1.2 定义输入/输出变量

定义输入/输出变量分为以下几个步骤。

(1) 双击打开 Stateflow Chart 模块，如图 27-4 所示。

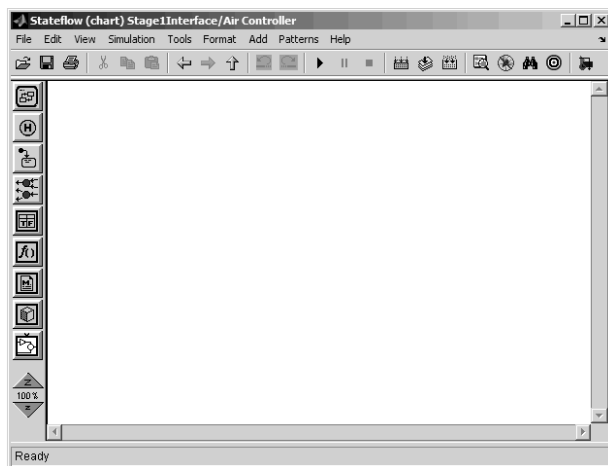


图 27-4 Stateflow Chart 模块编辑界面

(2) 增加输入/输出数据。

首先从菜单“Add”中选择“Data|Input from Simulink”，打开输入参数设置对话框

如图 27-5 所示。

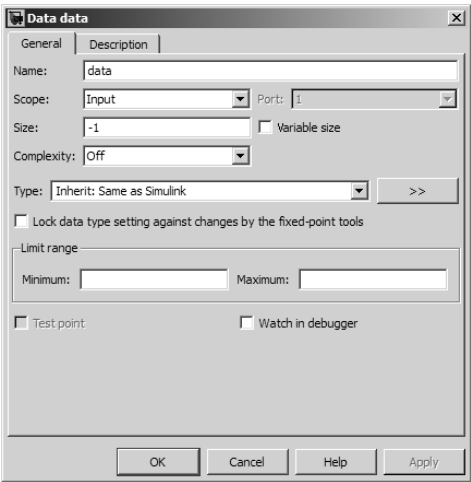


图 27-5 输入参数属性设置对话框

在属性对话框中设置相应的属性，部分属性默认值及含义如表 27-2 所示。

表 27-2 输入参数对话框默认值及含义

属 性	默 认 值	描 述
name	data	变量名称
Scope	Input	来自 Simulink 的输入变量
Size	-1	变量继承从 Simulink 中得到信号的大小
Complexity	off	变量中不含有复数值
Type	Inherit : Same as Simulink	继承 Simulink 中变量的类型

其次从菜单“Add”中选择“Data|Output to Simulink”，打开输出参数设置对话框，如图 27-6 所示，根据需要设置相应的参数。

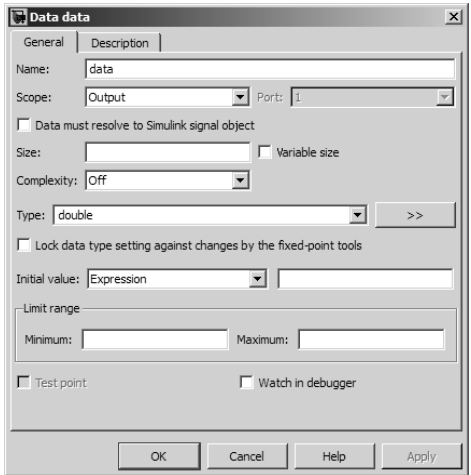


图 27-6 输出变量参数设置对话框



单击位于 Stateflow Chart 工具栏中向上箭头（如图 27-7 所示）返回 Simulink，得到具有输入/输出端口的模块，如图 27-8 所示。此时 Stateflow Chart 连接方式就与普通的 Simulink 模块一样，将它放入 Simulink 模型中，可组合完成仿真任务。

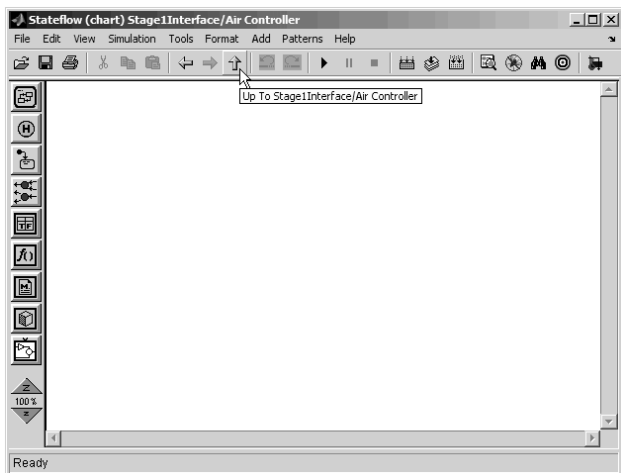


图 27-7 返回 Simulink

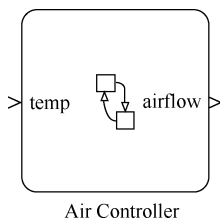


图 27-8 具有输入/输出变量的 Stateflow Chart 示例

27.2 飞机俯仰轴容错控制仿真

在飞机上为保证安全性，在某些关键部件上会采取冗余的方式，比如为控制飞机高度，采用升降舵，对升降舵采用几种方式的冗余。

图 27-9 所示为一个简化的具有冗余特性的升降舵系统。它由两个升降舵组成，一个在左边，另一个在右边。每一个升降舵都安装两个执行机构，在飞行过程中只有其中一个执行机构处于工作状态，另一个处于备份状态。这四个执行机构通过三个分离的液压线圈联系起来。

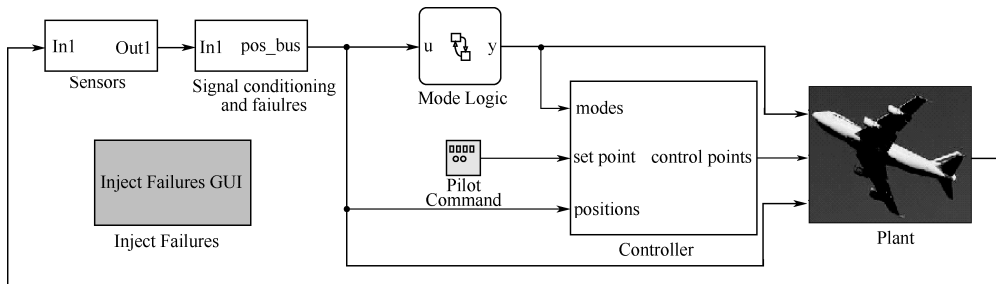


图 27-9 飞机俯仰轴容错控制系统

两套主要的飞行控制单元（PFCU）用于控制内外执行机构。

有两个控制律，在常态下，IO（输入-输出）控制律应用于左边和右边执行机构。如果失败，则采用直接联系控制律，该方法将减少功能并作用于左边内部执行机构和右边



内部执行机构。

在图 27-9 中采用了物理冗余的方案，当一个部件失效时，另外一个可以被激活，这一方面提高了系统的安全性，另一方面也给系统增加了复杂度和制造成本。

另外一种选择是功能备份，这种方式通过模型来得到系统额外的信息，虽然功能备份相对于硬件备份具有更小的成本，其仍然需要基于一定的硬件备份。

升降舵系统要求：

- 如果可以的话，左右舵应采用相同的控制律。
- 如果可以的话，尽量采用输入/输出控制律，而不是直接联系控制律。
- 未激活执行机构应处于待命状态。
- 如果对应的液压线圈等出现故障，则相应的执行机构应关闭或隔离。
- 控制器状态改变应在失效事件发生时产生。

上面自然语言描述的要求具有不完整、不一致和难以实现等缺点，这里考虑利用 Stateflow 实现其功能。

升降舵控制系统逻辑实现如图 27-10 所示。

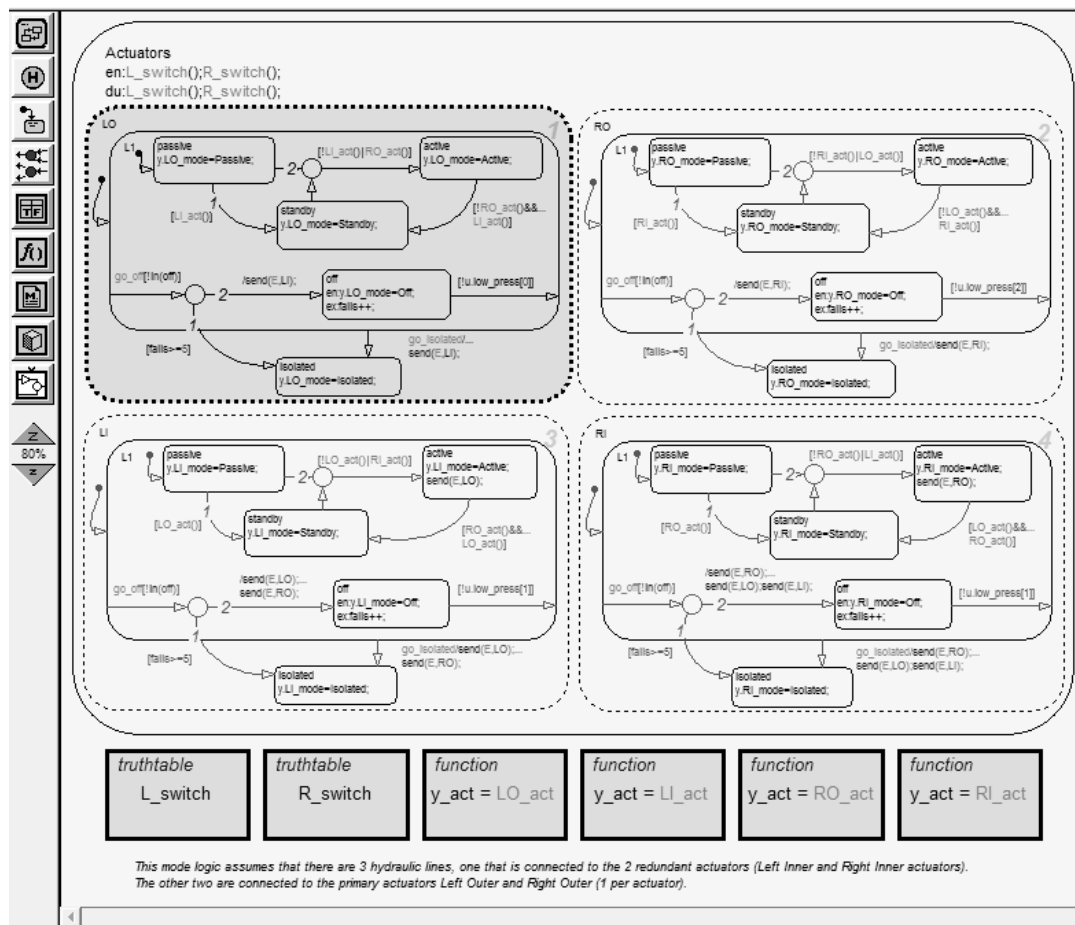


图 27-10 升降舵控制系统逻辑实现

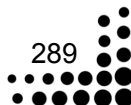




图 27-10 左上角模块给出了包含 IO 控制律左执行机构模块变换逻辑，其他三个状态转换表也是一致的。但是转换的条件与变量是不同的。四个状态转换表的执行顺序如下：LIO 模块首先执行，其次是 RIO 模块，然后是 LDL 模块，最后是 RDL 模块。这种执行顺序对决定恢复逻辑非常关键。

当 PFCUs 加电时，状态转换表首先进行进入转换（通常是默认转换，而且箭头末端有实心黑点），从图中可以看出，此时 LIO_mode 被设置为 2。相似的，其他三个将变量 RIO_mode, LDL_mode 和 RDL_mode 都设置为 2。从这个被动状态可以进入主动状态。

LIO 进入主动状态的条件为，LDL 模块不是主动状态或者 RIO 模块是主动状态，用表达式表示为 $[!LDL_act()|RIO_act()]$ 。否则，LIO 模块将进入“standby”状态。第一个条件中“ $!LDL_act()$ ”用于确定升降舵中至少有一个主动执行机构。第一个条件中“ $RIO_act()$ ”用于保持对称，即如果右升降舵由 IO 模块运行，则左升降舵也应该由 IO 模块执行（在允许的情况下）。这种转换机制同样适用于其他三个执行机构。在默认情况下，无故障发生时，LIO 和 RIO 将进入“active”状态，而 LDL 和 RDL 模块将进入“standby”状态。

执行机构重要的一点是采用了分层的结构。例如，左上的 LIO 模块，分层允许从其他状态进入“isolated”状态。

当 LIO 模块变换为“off”或者“isolated”状态时，相应的事件将会广播到其他模块，这保证其他模块处于正确的状态。

考虑到 LIO 模块进入“isolated”状态。在这个转换中，事件“E”广播到 LDL 模块，并通过检查“ $[!LIO_act()|RDL_act()]$ ”，通过检查结果使 LDL 模块由“standby”状态转换为“active”状态。然后 LIO 模块完成转换到“isolated”，其次 RIO 模块也进行检测，由于“ $[!LIO_act() \& \& RDL_act()]$ ”不满足，而处于“active”状态。LDL 检测“ $[RIO_act() \& \& LIO_act()]$ ”继续保持“active”状态，RDL 模块由检测“ $[!RIO_act()|LDL_act()]$ ”保持“active”状态。由于它进入“active”状态，并广播事件 E 到 RIO 模块，使得它进入“standby”状态。

升降舵建模包含一个受到以下三个因素影响的质量块：

- 由执行机构给出的活塞驱动。
- 受到与升降舵偏移成正比的风载影响。
- 受到与升降舵运动方向相反的摩擦阻力影响。

活塞力由液压执行机构产生，由伺服阀、随动阀和一个圆柱体组成。伺服阀控制流入圆柱体的油流速，该流速大小由反馈控制程序给出。当执行机构处于“standby”状态时，伺服阀的活塞被处于“active”状态的执行机构掩盖。为不使处于“standby”状态的活塞影响处于“active”状态的活塞，随动阀用于阻断处于“standby”状态活塞油的流动。

27.3 本例小结

本例在学习如何基于 Stateflow 建立有限状态机的基础上，实现了飞机俯仰轴容错控制仿真，通过本例学习，对基于 Stateflow 建立仿真系统有了初步的了解，通过后面三个实例可进一步加深。



第 28 例 汽车电动车窗升降控制仿真

汽车电动车窗是指通过电动机或其他工具使车窗实现自动升降的汽车车窗。电动车窗一方面为实现便利性，需要较高的机动性；另一方面为保证安全性，在上升过程中遇到障碍物时需要能够及时识别并停止运动，这些控制逻辑可以通过 Stateflow 来实现。

本例在介绍 Stateflow 运行机理的基础上，以汽车车窗为对象，介绍建立俯车窗自动升降控制逻辑并对其仿真。通过本例学习，需要掌握以下两点：

- Stateflow 和 Simulink 运行机理。
- 基于 Stateflow 实现电动车窗过程。

28.1 Stateflow 运行机理

在前面介绍 Stateflow 基本模块及建立有限状态机的基础上，为进一步让读者了解 Stateflow，这里对 Stateflow 运行机理进行介绍。

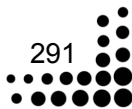
28.1.1 有限状态自动机与 UML 状态图理论概述

1. 有限状态机基本概念

当系统行为可以被分解成状态有限且不重叠的程序块时，系统表现出状态行为。可通过有限状态自动机（FSM）建模状态行为，FSM 可分为确定型和非确定型，非确定型可以转化为确定型。确定型 FSM 在每个输入上，可从当前状态转换到有且仅有的一个状态。因为 Stateflow 状态图的特性，主要介绍确定型 FSM。

一个确定型 FSM 包括有穷状态集合 Q 、有穷输入符号集 Σ 、转换函数 $\delta(Q \times \Sigma \rightarrow Q)$ 、属于 Q 的初始状态 q_0 和终结状态或接受状态集 F （ Q 的子集）。通常用 DFA 的五元组 $A = (Q, \Sigma, \delta, q_0, F)$ 表示确定型 FSM。自动机从初始状态 q_0 起，逐一读入由输入字母表 Σ 元素构成的输入串的字母，根据当前状态、输入字母和转换函数 δ 来决定自动机的下一步状态。若输入串结束时，自动机处于终结状态集合 F 的某个状态，表示自动机接受该字符串；否则表示不接受。

经典 FSM 可以解释为 Mealy 型和 Moore 型。Mealy 型的输出与现行状态的输入相关，而 Moore 型的输出只与现行状态相关。另外，执行动作需要花时间，因此状态机在两个模式间切换：空闲监听和下一事件到达响应事件。若系统正忙于处理前一事件，对于出





现的更高优先级事件有抢先和非抢先两种处理方式。抢先处理会使单个状态机内部引入并造成互斥等问题，而非抢先会存储新事件并继续处理前一事件，使系统以离散、不可分的 RTC 处理各事件。

2. UML 状态图概述

Stateflow 中状态图语法类似于 UML 层次式状态机 (HSM)。UML 状态图是具有 Mealy 和 Moore 两种自动机特性的扩展状态机。在状态图中，动作符合 Mealy 自动机，与系统状态和触发事件相关；状态图提供的可选进入/退出动作与状态而非转换相关，与 Moore 自动机相同。

1) 层次状态机与行为继承

状态图对于经典状态机最重要的改进就是引入层次式嵌套。若系统属于被嵌套 S11 子状态，则系统也属于 S1 超状态。若状态 S11 没有处理该事件，将在状态 S1 中处理该事件。状态嵌套可分为多层，事件处理规则可递归应用于任何嵌套层次。UML 规范定义了一个顶层组合状态作为 HSM 的根。因为未处理的事件可被高层状态处理，被嵌套子状态而非转换关系，与 Moore 自动机相同。

2) UML 规范

UML 状态图中每个状态有可选的进入状态和退出动作，该动作只与状态相关，与转换无关。进入和退出动作提供了初始化和清除的方法，类似于 OOP 中类的构造函数和析构函数。进入动作通常被用来标识状态，如同类构造函数确定构造对象的标识。与类构造函数的调用类似，进入动作从最外层状态顺序执行，而退出动作按相反顺序执行。但进入和退出动作也有与构造和析构函数不同之处。改变类对象的标识在销毁之后再重构，即使初始与最后的对象继承于同一父类，而状态图中标识变化时状态转换。某个状态包含子状态 A 和 B，从 A 状态到 B 状态仅执行 A 状态退出动作和 B 状态进入动作，不涉及父状态动作。

内部转换指的是某事件只引起某些内部动作而不导致状态改变。除了内部转换，每个状态转换都会引起状态图退出源状态和进入目标状态配置。状态转换的动作顺序是：源状态的退出动作、与转换相关的动作和目标状态的进入动作。首先从现行状态退出所有被嵌套状态，一直到源与目标状态的最小公共祖宗状态（但不退出此状态）。执行完转换动作后，从退出动作停止层次依次执行进入动作至目标状态。对于组合状态，其子状态按照规定的转换或历史机制递归执行进入动作。

3) 状态图与流程图的区别

状态机执行针对现实触发的动作，而流程图不需要显示触发。流程图中节点的转换依赖于动作的完成。相比于状态图，流程图颠倒了图中顶点和弧的意义，状态图中的处理与弧关联，而流程图中的处理与顶点关联。流程图通过使用连接节点和转换来构造模型逻辑模式。连接节点为可选转换路径提供分支判断，可用此表示判断和循环逻辑。无状态的流程图不能保存活动状态，通常起始于默认转换，终止于某个无流出转换的终结节点。

28.1.2 Stateflow 机制分析与实现思路

1. 事件驱动机制

Stateflow Charts 对某个事件响应，并按照循环的方式执行，如图 28-1 所示。因为一个 Chart 在一个单线程中运行，基于事件的动作对该动作具有原子性。由事件引起的活动在执行返回到接受事件前进行的活动之前完成。只要事件触发产生动作，该动作只要不被提前返回中断就将完成。Stateflow Charts 通常以自顶向下的层次结构处理事件，包括以下两步。

- (1) 执行活动状态的 during 和 on event_name 动作。
- (2) 检测子状态的有效转换。

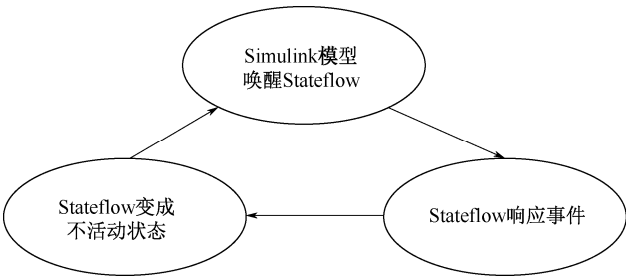


图 28-1 事件驱动机制

若事件的接受者处于活动状态则执行时间，否则不执行。在广播完事件后，广播者根据引起事件的动作描述类型执行提前返回逻辑。

Stateflow Charts 执行包括三个阶段。当 Simulink 模型首次触发 Stateflow Charts 时，Charts 不活动且无活动状态。在 Charts 执行处理完初始触发事件后，它将控制转换使自己进入睡眠。Charts 在下一个 Simulink 触发事件从睡眠变成活动，三个阶段的状态描述如表 28-1 所示。

表 28-1 Stateflow Charts 的状态描述

阶 段	描 述
Inactive 不活动	Charts 没有活动状态
Active 活动	Charts 有活动状态
Sleeping 睡眠	Charts 没有活动状态，但没有事件待处理

2. 状态的进入、执行和退出过程

1) 进入状态执行过程

当进来的转换跨越或结束于状态边界，或属于活动态的 parallel 子状态，当前状态将变成活动时执行 entry 动作，进入状态的执行步骤如下。



- (1) 如果该状态的父状态不活动, 先为其父状态执行步骤 1 至 3。
- (2) 若该状态是 parallel 状态, 如果其年长的兄弟状态处于活动, 为其兄弟从步骤 1 开始执行。其中, 可以显式或隐式地排序 parallel 状态。
- (3) 将状态标记为活动, 并且执行 entry 动作 (如果有)。
- (4) 如果该状态有子状态, 则进入子状态: 若状态不含历史节点, 执行该状态的默认流程路径; 若状态含历史节点并且存在一个在最近的 Charts 初始化后的活动子状态, 执行该子状态的进入动作; 若状态含有 parallel 子状态, 按照子状态顺序, 为每个状态执行步骤 1 至 4。
- (5) 若该状态是 parallel 状态, 为其所有年幼兄弟执行进入操作。
- (6) 若转换路径父状态与当前状态父状态不同, 为该状态的直接父状态执行步骤 5、6。
- (7) Charts 进入睡眠。

2) 活动状态执行过程 (当状态变为活动执行)

- (1) 执行外部流程图集合; 若引起状态转换则终止执行 (parallel 无此状态)。
- (2) 按照状态标签的顺序, 执行 during 动作和有效的 on event_name 动作。
- (3) 执行内部流程图集合。若该步骤未引发状态转换, 活动子状态从步骤 1 开始执行。同时, parallel 状态按照它们变成活动的顺序进行执行。

3) 退出活动状态过程

当出去的转换跨越或产生于状态边界, 以及属于活动状态的子 parallel 状态时, 当前状态将变成不活动执行退出过程, 具体步骤如下。

- (1) 从最后执行进入状态的兄弟 parallel 状态开始, 逆序执行。
- (2) 若该状态有活动子状态, 按照子状态变成活动的相反顺序执行退出状态。
- (3) 执行任意退出动作, 并且将状态记为不活动。

3. Simulink/Stateflow 形式语言描述

一个 Simulink 模型表示了输入、状态、输出之间时间依赖的数学模型, 可被定义为 $SL = (D, B, C, L)$, 其中:

- 有型变量组成的有限集合 D 划分为输入、状态、辅助和输出变量 D_I 、 D_S 、 D_A 和 D_O 。
- Simulink 模块组成有限集合 B , 每个模块包括输入、输出和局部变量。输入和输出变量与输入、输出端口关联。
- 一个有序关系 $C \in B \times B$ 表示模型间的关系。一个关系 $c = (b, b') \in C$ 连接了输出端口 b 和输入端口 b' , 同时它表示相对应变变量 b 和 b' 的数据传递。
- 一个函数 $L: C \rightarrow D_A$, 关联了一个独立的辅助变量到每个连接上。

按照类似方法, 一个 Stateflow 流程图可被定义为 $SF = (V, E, S, TR, T)$, 每部分元素定义如下:

- 由有型变量组成的有限集合 V 被划分为输入、局部、输出变量 V_I, V_L, V_O 。



- 事件组成有限集合 E 被划分为输入、局部和输出变量 E_I, E_L, E_O 。
- 由状态构成的有限集合 S 被划分为原子、AND 和 OR 状态。活动态 AND 状态其子部件全将是活动态。活动态 OR 状态，有且仅有一个子部件变为活动态。每一个状态 $s \in S$ 被标记为一系列动作 AS 的有限集合。一个动作是一个非输入变量的赋值或一个事件广播。动作集合 AS 被划分为有序动作集合。一个动作是一个非输入变量的赋值或一个事件广播。动作集合 AS 被划分为有序动作集合：进入 s 、退出 s 之前以及 s 处于活动态的动作集合 $\text{entry}(s)$ 、 $\text{exit}(s)$ 和 $\text{during}(s)$ 。
- 一个关系 $TR \in S \times S$ 表示状态间的层次组合。图 (S, TR) 是一颗树。树边 $(s, s') \in TR$ 表示 s' 是 s 的子状态，在 S 中只有原子状态才是树的叶节点。一个图表的状态活动被表示为一棵树 $(S_{\text{Act}}, TR_{\text{Act}})$ ，其中 $S_{\text{Act}} \in S$ ， $TR_{\text{Act}} \in TR$ 。 TR_{Act} 中的树边表示组合状态和它们子状态的关系，其值可取 AND 或 OR。
- T 是转换的有限集合。一个转换 $t \in T$ 是一个元组 $t(\text{tuple } t) = (s, s', e, \phi, \text{act}_{\text{condition}}, \text{act}_{\text{transition}})$ 。其中 $s, s' \in S$ 分别是源状态和目标状态， $e \in E$ 是一个使能事件， ϕ 是对变量 V 的断言逻辑， $\text{act}_{\text{condition}}$ 是当 ϕ 为真时执行的条件动作，而 $\text{act}_{\text{transition}}$ 是进入状态 s' 前执行的转换动作。

28.2 汽车电动车窗升降控制仿真

随着汽车电子技术的发展，电动车窗系统在汽车上的应用也越来越广泛，比如说车窗升降、车门开关等。这些电子系统在为驾驶和乘坐者带来便利的同时，也带来了隐患，如果电子系统出现故障，将会出现危险甚至危及人的生命。

由于车窗安全隐患的存在，因此在生产之前，需要进行仔细的设计与生产。

下面以乘客边窗为例介绍其设计，边窗设计要求有以下几点：

- 车窗完全打开或者完全关闭需要在 4s 内完成。
- 指令给出后，车窗需要在 200ms 以后 1s 以内开始运动。
- 当出现障碍物时，施加的力必须小于 100N。
- 碰到障碍物时，下降 10cm。

在以上要求中，重要的是后面两点，这两点保证车窗运动对人体不产生危险。

根据需求建立汽车电动车窗升降控制系统，如图 28-2 所示。

由图 28-2 可以看出系统分为指令输入、车窗动力学与控制，以及逻辑控制三部分。

下面对这三部分内容分别予以介绍。

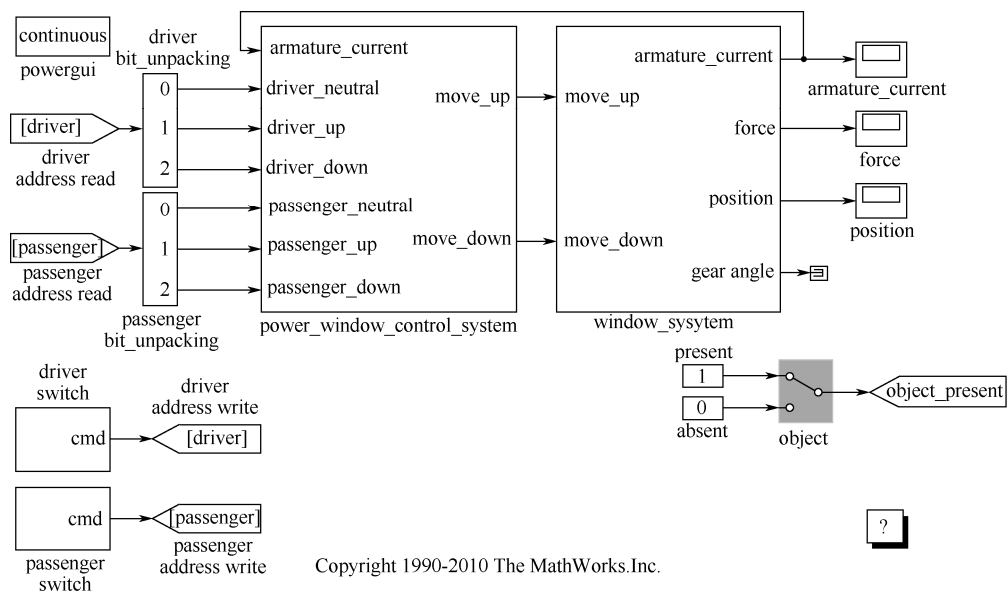


图 28-2 汽车电动车窗升降控制系统

28.2.1 指令输入部分

电驱动的指令输入模型如图 28-3 所示，乘客指令输入部分与此类似。图 28-3 (b) 是 (a) 中 switch 的内部结构。

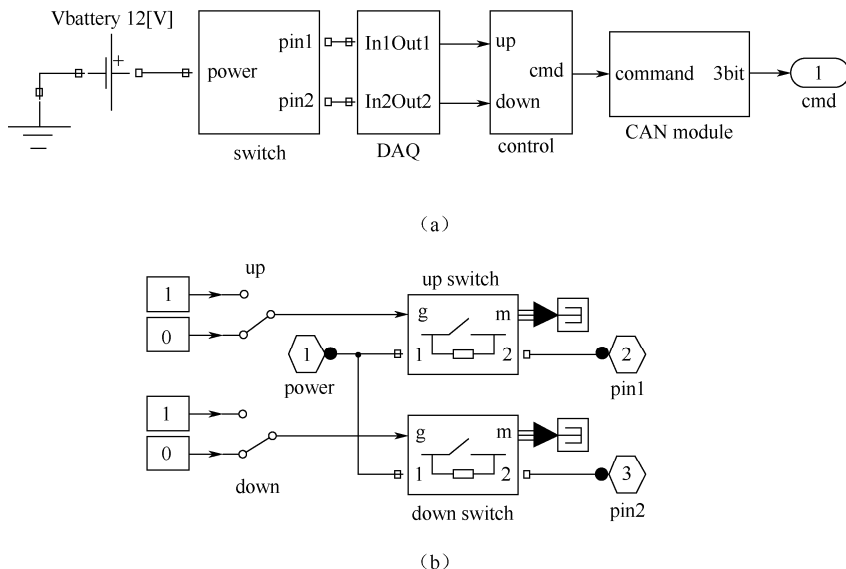


图 28-3 指令输入模块

由图可以看出，电动车窗有两个输入：

(1) 乘客输入。

乘客输入没有设置时：

上升：乘客控制使车窗产生上升命令。

下降：乘客控制使车窗产生下降命令。

(2) 驱动输入。

驱动输入没有设置时：

上升：驱动控制使车窗产生上升命令。

下降：驱动控制使车窗产生下降命令。

此外为保证安全和性能，还需要以下两个状态输入：

(1) 是否到达车框的顶部或底部。

0：车窗未到达车框顶部或底部。

1：车窗位于车框顶部或底部。

(2) 是否遇到障碍物。

0：车窗未遇到障碍物。

1：车窗遇到障碍物。

28.2.2 车窗动力学与控制部分

车窗动力学与控制部分如图 28-4 至图 28-6 所示。其中图 28-5 为控制力矩生成部分，图 28-6 为车窗动力学部分，图 28-4 为两者的组合。

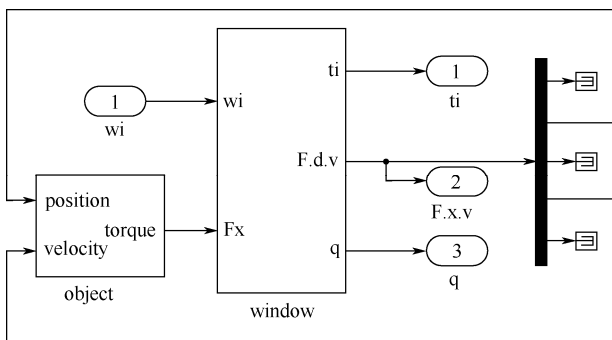


图 28-4 车窗控制力矩及车窗动力学组合

由图 28-5 可以看出，车窗控制力矩主要采用 PD 控制律，设控制力矩为 T ，比例系数为 k_p ，微分系数为 k_d ，距离和速度分别为 d 和 v ，则控制力矩如式 (28-1) 所示。

$$T = k_p d + k_d v \quad (28-1)$$

在没有遇到障碍物及车窗到顶指令之前，可按照此力矩驱动车窗进行运动，但由于安全性等问题，需要额外的控制逻辑来影响，这部分在 28.2.3 中予以介绍。

由图 28-6 可以看出，车窗动力学模型基于 SimMechanics 工具箱实现，基于 SimMechanics 工具箱能够更好地实现机械对象动力学仿真，可以看出基于 MATLAB 不用



工具箱之间的组合，能够实现多物理领域对象的仿真。

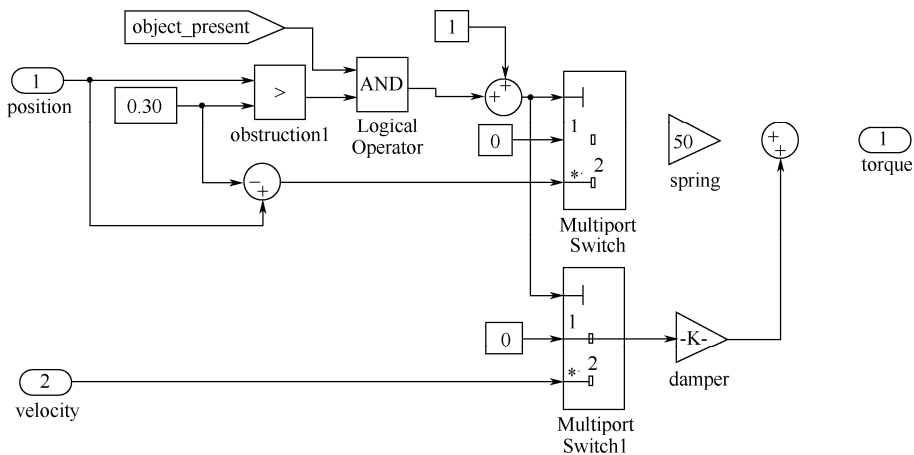


图 28-5 车窗控制力矩模型

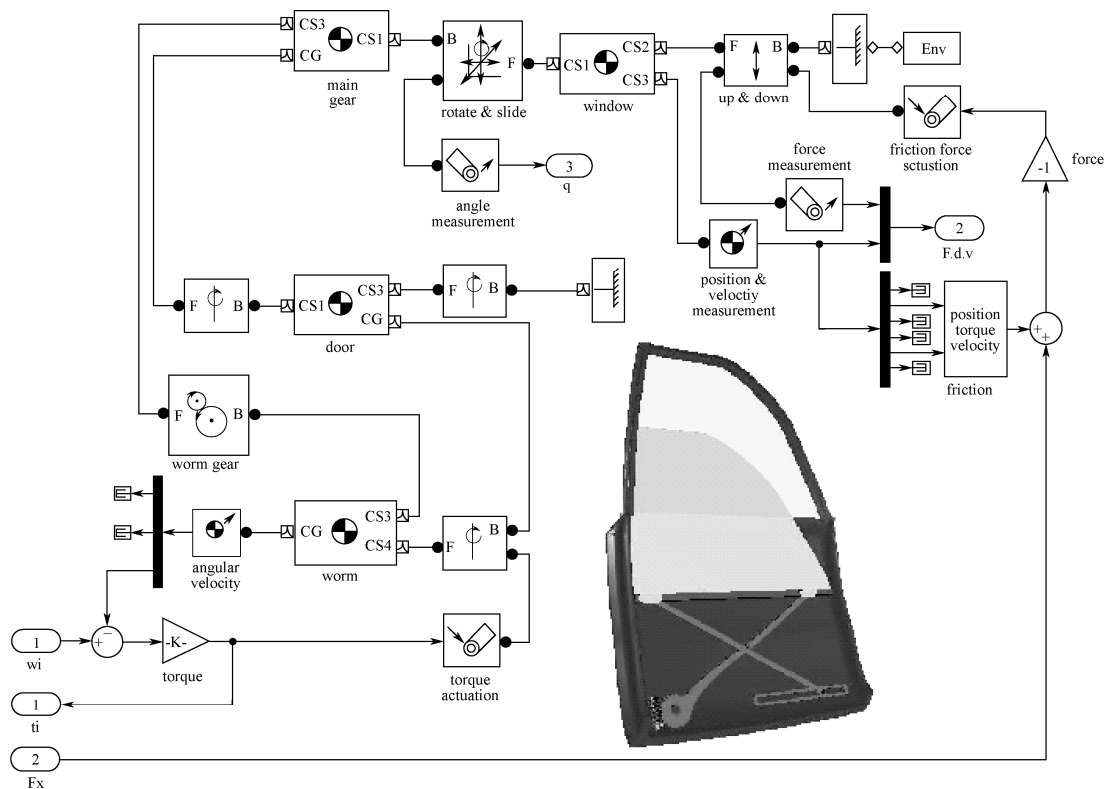


图 28-6 车窗动力学模型

28.2.3 车窗控制逻辑部分

车窗控制逻辑如图 28-7 所示。

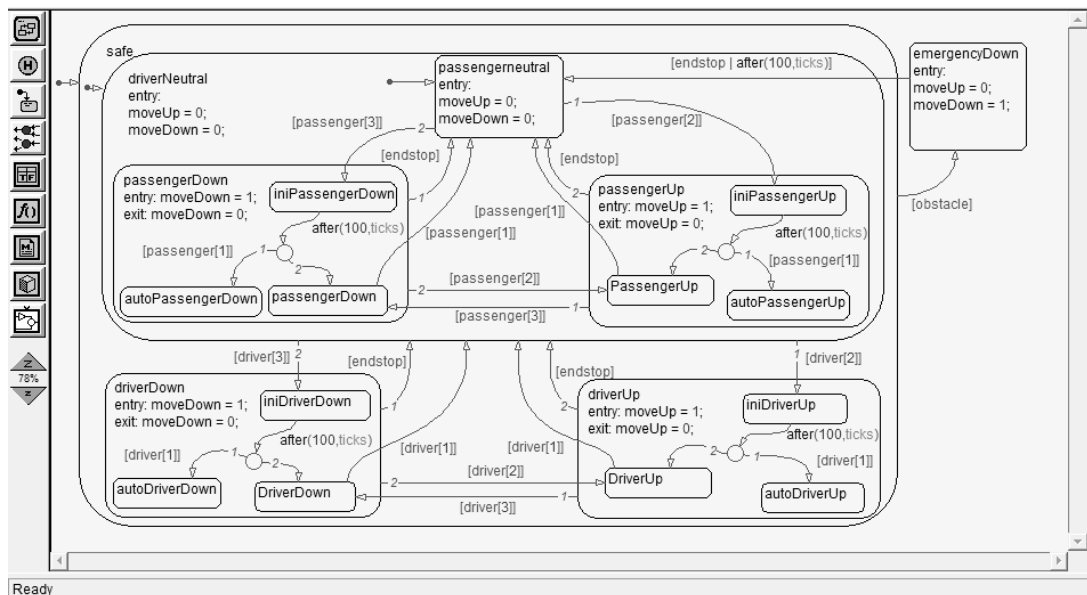


图 28-7 车窗控制逻辑图

图 28-7 给出电动车窗关系框图。图中有四个对象：驱动、乘客、车窗和控制器。驱动和乘客都可以发出指令使车窗上升或下降，驱动或乘客发出的指令都是首先发给控制器，控制器根据输入指令和控制逻辑来操控车窗上升或不动。此外，车窗当前的状态需要被监视，以了解车窗当前是处于完全打开、完全关闭或车窗与车框之间是否存在障碍物。在图中，控制器由一个圆圈表示，这也表示一个过程，对于简单的过程，控制输出可直接生成，对于复杂的过程，可能需要通过分析组合逻辑或连续结构来推断由输入得到的输出信号。

图中包含三个过程，前两个用于验证驱动和乘客的输入，比如说，车窗完全打开，这个时候再输入“Window Up”命令也是无效的，最后一个过程用于检测是否存在障碍物以及车窗应该上升还是下降。

28.2.4 仿真结果

车窗上升时遇到障碍物时的控制结果如图 28-8 所示，由图可以看出系统能够很好地辨识障碍物并保证足够的安全性。

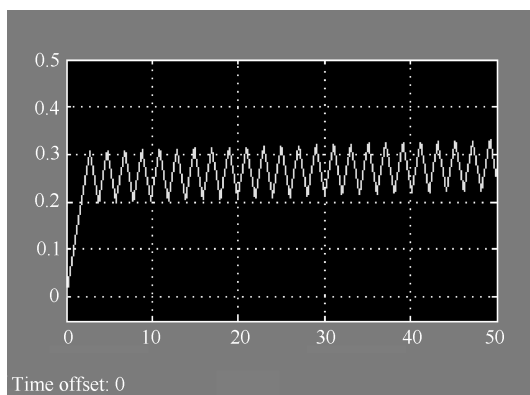


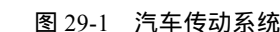
图 28-8 车窗升降控制结果

28.3 本例小结

本例在学习 Stateflow 运行机理的基础上，对汽车电动控制进行了建模与仿真，通过本例学习可对基于 Stateflow 仿真运行机理有更加深刻的了解。

本例通过对汽车传动系统进行建模与仿真，读者需掌握以下两部分内容：

- 汽车传动系统如图 29-1 所示, 由图可知系统包含以下几部分: 变速箱、驱动设备部分、逻辑部分、发动机二维表、力矩转换器和引擎。下面分别针对这几部分系统予以介绍。





29.1 变速箱

汽车上的自动变速器必须完成很多任务：如果汽车位于超速挡（在四挡变速器上），变速器将根据车速和节气门踏板位置，自动选择齿轮；如果缓慢加速，则换挡速度会比在节气门全开状态下加速的换挡速度要低；如果把加速踏板踩到底，变速器将降到下一个较低挡。

自动变速器和手动变速器完成同样的事情，但它们完成的方式完全不同。与手动变速器一样，自动变速器的主要工作是让发动机在较窄的转速范围下运动，并且提供较宽的输出速度范围。

如果没有变速器，汽车将会只有一种传动比，这种情况下只能选择让汽车以所需的最大速度行驶的那种传动比。因此，变速器使用齿轮，以便更有效利用发动机的扭矩，从而保持发动机在核实的转速下运行。

手动变速器和自动变速器之间的关键不同在于：前者将不同组的齿轮分别锁定到输出轴，以得到各种传动比，而在自动变速器中，同一组齿轮就可得到所有不同的传动比，自动变速器则是通过行星齿轮组来实现这一功能的。

汽车自动变速箱模型如图 29-2 所示，由图可以看出变速箱主要由行星齿轮组和离合器组成。下面分别针对这两部分进行介绍。

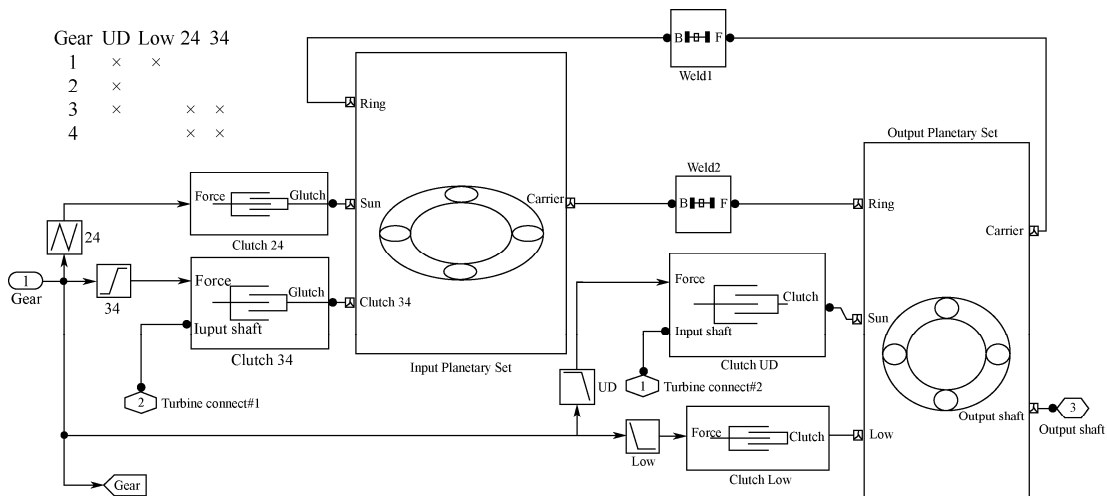


图 29-2 汽车变速箱

29.1.1 行星齿轮组

当我们分解自动变速器以了解其内部结构时，会发现其在相当小的空间内容纳了各种各样的部件。除了其他部件外，还可以看到：

- 一套精致的行星齿轮组。



- 一组钢带，用于固定齿轮组的部件。
- 一组三个湿盘离合器，用于固定齿轮组的其他部件。
- 一套液压系统，用于控制离合器和钢带。
- 一个大型齿轮泵，用于运送变速器液力传动油。

行星齿轮组的大小与甜瓜相仿，它产生变速器所能生成的所有不同传动比。变速器内的其他所有部件都是为了帮助行星齿轮组完成此工作。自动变速器包含两套完整的行星齿轮组，它们组合成一个部件。

所有行星齿轮组都有三个主要部件：

- 太阳轮。
- 行星齿轮和行星齿轮的齿轮架。
- 齿圈。

每个部件可以作为输入、输出，也可以保持不动。当各种部件担任不同角色时，可相应得到齿轮组的某一传动比。

变速器的一个行星齿轮组包括一个 72 齿的齿圈和一个 30 齿的太阳轮。通过该齿轮组，可以得到很多不同的传动比。

另外，将其中任何两个部件锁定在一起都会将整个装置锁定在 1:1 齿轮减速比。请注意，上面列出的第一个传动比是减速挡——输出速度比输入速度慢，第二个是超速挡——输出速度比输入速度快。最后一个又是减速挡，但输出方向相反。从这个行星齿轮组还能得到其他几种传动比，不过这几种传动比与自动变速器相关。

因此，不需要啮合或脱离任何其他齿轮，这组齿轮就可以产生所有不同的传动比。两套这样的齿轮组排成一行，就可以得到变速器需要的四个前进挡和一个倒挡。

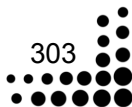
29.1.2 离合器和制动带

在变速器中，啮合超速挡后，连接到液力变矩器外壳的轴（通过螺栓固定到发动机的飞轮）会通过离合器连接到行星架。较小的太阳轮空转，较大的太阳轮被超速挡制动带固定。没有任何部件连接到涡轮，仅有的输入来自变矩器外壳。

为了将变速器转到超速挡，必须通过离合器和制动带连接和断开许多部件。行星架通过离合器连接到液力变矩器外壳。较小的太阳轮通过离合器从涡轮断开，使其能够空转。较大的太阳轮通过制动带固定到外壳，使其无法旋转。每次换挡都触发一系列类似的操作，只不过啮合与脱离的离合器和制动带不同。

在本变速器中，有两幅制动带，变速器中的制动带实际上是钢带，缠绕在齿轮系的截面上，连接到外壳，它们通过变速器壳内的液压缸驱动。

变速器中的离合器有一点复杂。在本变速器中有四个离合器，每个离合器都是由增压过的液压油驱动，这些液压油进入离合器内的活塞中。弹簧确保当压力下降时，离合器松开。





29.2 引擎

引擎如图 29-3 所示。

引擎以节流阀 (Throttle) 为输入，将输出传递给力矩转换器。

$$I_{et} \dot{N}_e = T_e - T_t$$

式中 N_e 为引擎速度， I_{et} 为引擎和叶轮的转动惯量， T_e 和 T_t 分别为作用于引擎和叶轮的力矩。

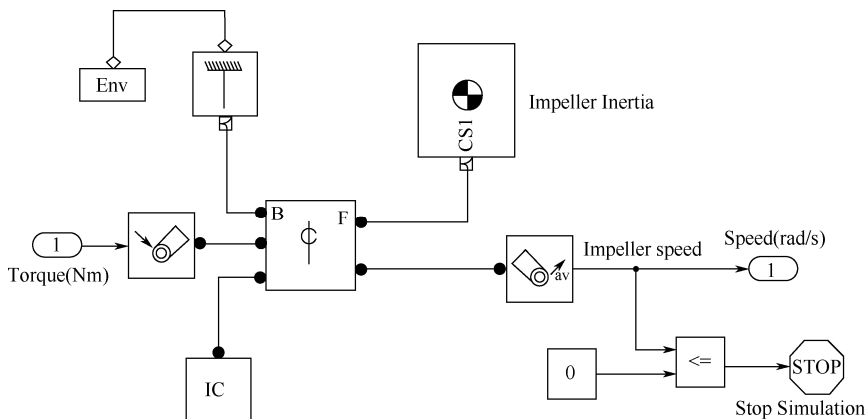


图 29-3 汽车引擎模型

29.3 液力变矩器

液力变矩器是以液体作为工作介质，利用液体的动能进行能量传递的装置，由德国工程师费丁格尔发明的机械，最早用于船舶以解决动力性能不匹配的问题。液力变矩器大量用于车辆并取得成功，液力变矩器主要优点如下。

1. 具有良好的自适应性

当外载荷增大时，液力变矩器能使车辆自动增大牵引力，同时车辆自动减速，以克服增大的外载荷。反之，当外载荷减小时，液力变矩器又能使车辆自动地减小牵引力，提高车辆速度。因此液力变矩器极大地改善了车辆的牵引特性。

2. 具有良好的过载保护性

液力变矩器将发动机的工作点限定在一定的工作范围内，无论外载荷多大，而施加在发动机上的最大力矩是固定值，这避免了发动机因外载荷突然增大而熄火。

3. 提高车辆的使用寿命

由于液力传动的介质能够吸收并减小来自发动机及外载荷的振动与冲击，这就是液力传动的滤波作用，因而可以提高车辆的使用寿命。

4. 提高了车辆的通过性

在使用了液力传动之后，可以使车辆以爬行的速度行驶，可以不破坏路面，使附着力不降低。

5. 提高了车辆的舒适性

采用液力传动后，车辆的起步平稳，加速均匀，并在较大范围内无极变速，可以吸引和减小振动与冲击，从而可以提高车辆的舒适性。

6. 简化了车辆的操作

液力变矩器本身就是一个自动无极变速器，使用了液力变矩器之后，发动机的动力范围得到了扩大，变速器的挡位数可以减小，简化了车辆的操作。

7. 车辆可以带载启动

车辆安装了液力变矩器后，发动机启动时，车辆施加给发动机的载荷比较小，可以带载启动。

液力变矩器的主要缺点是传动效率较低，且结构较为复杂，制造成本较高。

液力变矩器模型如图 29-4 所示。其输入输出可表示为引擎速度和涡轮速度的函数，在这个例子中，能量流认为是由转换器流向涡轮。

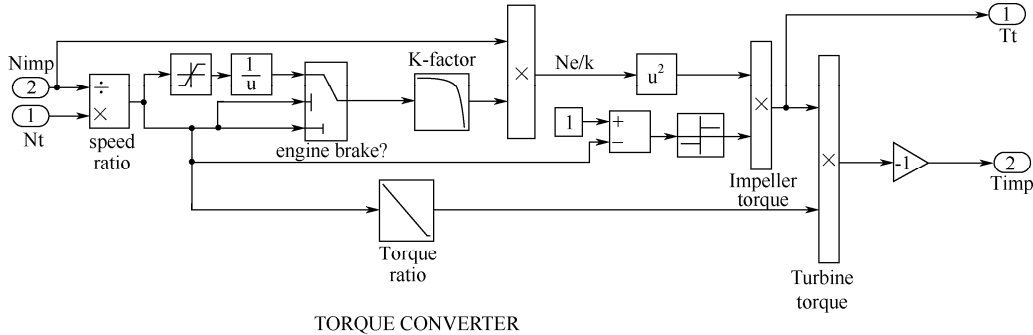


图 29-4 液力变矩器

$$T_i = \frac{N_2^2}{K^2}$$

$$K = f_2 \frac{N_{in}}{N_e} = K - \text{factor}$$

式中 N_{in} 为涡轮速度，即力矩转换器输出和变速器输入。



$$R_{T_Q} = f_3 \frac{N_{in}}{N_e} = \text{力矩比率}$$

传递模型通过静态齿轮比率并基于小变速时间来实现。

$$R_{T_R} = f_4(\text{gear}) = \text{传递比率}$$

$$T_{out} = R_{T_R} T_{in}$$

$$N_{in} = R_{T_R} N_{out}$$

式中 T_{in} 和 T_{out} 分别为传递输入力矩和输出力矩， N_{in} 和 N_{out} 分别为传递输入速度和输出速度。

29.4 驱动系统及设备

汽车本体驱动动力学如下：

$$I_v \dot{N}_w = R_{fd} (T_{out} - T_{load})$$

式中 I_v 为汽车惯量， N_w 为涡轮速度， R_{fd} 为最终驱动比率， T_{load} 为负载力矩。

负载力矩包含路面负载和刹车力矩，路面负载还包含摩擦和空气损失。

$$T_{load} = \text{sgn}(\text{mph}) (R_{load0} + R_{load2} \text{mph}^2 + T_{brake})$$

式中 R_{load0} 和 R_{load2} 分别为摩擦和空气阻力系数， T_{load} 和 T_{brake} 分别为负载和刹车力矩，mph 为汽车线性速度。

根据此建立汽车驱动系统动力学模型，如图 29-5 所示。

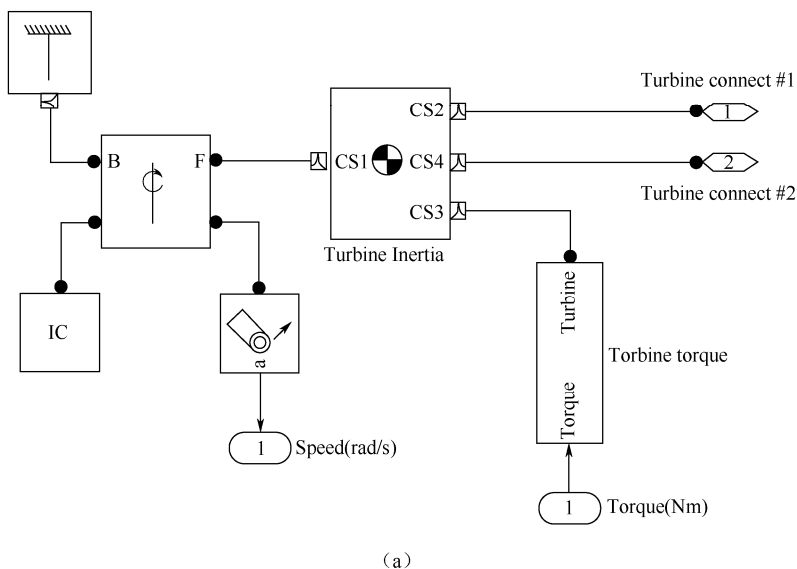
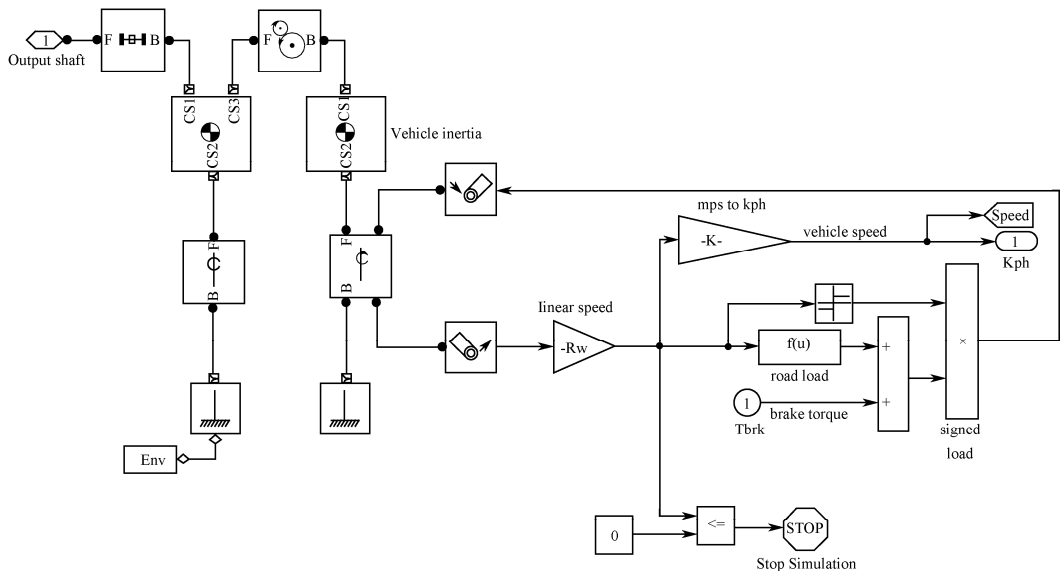


图 29-5 汽车驱动系统模型



(b)

图 29-5 汽车驱动系统模型（续）

29.5 发动机表格

发动机的工作过程是一个复杂的过程、运行机理非常复杂，难以用解析的表达式来准确描述其运动状态。目前基本上都是利用测功试验的方法获得数据并采用辨识的方法建立模型。在 MATLAB 中一般是建立表格并利用插值的方法来获得发动机数据。下面介绍 MATLAB 中表格插值技术。

表格插值是一种静态建模技术，它常用于建立系统动态方程，如方程系数；还可以用于计算大量系统的结果估计，常用的方法有线性插值法和三次样条插值法，线性插值可分为等间距和不等间距插值法。下面对等间距插值、不等间距插值和三次样条插值原理进行介绍。

1. 等间距线性插值

当输入 X_i 属于闭区间 $[X(0), X(0) + (N-1)\Delta X]$ ，较小插值点 $L = \lfloor (X_i - X(0)) / \Delta X \rfloor$ ，在索引 L 和 $L+1$ 插值点间进行插值得到输出 $y = y(L) + [y(L+1) - y(L)](X_i - X(L)) / \Delta X$ 。

2. 不等间距线性插值

将 x 、 y 轴上插值点保存在长度为 N 的数组中 $\text{arr_x}[N]$ 、 $\text{arr_y}[N]$ ，下标以 0 索引。输入值 X_i 满足 $\text{arr_x}[N] < X_i < \text{arr_x}[N-1]$ ，同时设索引变量 $L=0$ ， $U=N-1$ 。

二分法重复以下步骤确定插值点对 ($U=N-1$)：

- 置当前索引值 $i = \lfloor (L+U)/2 \rfloor$ 。



- 若在索引 i 处的插值点大于输入 X_i , 设 $U=i$, 否则 $L=i$ 。

循环以上步骤, 得到以 L 为修正后插值点间隔内较小插值点索引值。

按照下式在索引为 L 和 $L+1$ 的插值点间进行插值:

$$y=y(L)+[y(L+1)-y(L)](X_i-X(L))/(X(L+1)-X(L))$$

- 三次样条和多维表格插值。

三次样条插值运用三阶多项式估计两个插值点间的函数, 可获得比线性插值法更平滑的插值函数。在两个插值点的闭区间, 函数及其一阶、二阶导数均连续。但是使用此方法需要更多执行时间以及更大内存。该算法在仿真初始化时, 首先对每个插值点间隔确定多项式系数。仿真运行时, 先使用等间距或不等间距插值法确定包含输入值的插值间隔, 然后使用之前获得的系数计算多项式。

之前讨论的都仅涉及一个输入变量, 多输入插值函数中, 每个输入均有一个插值点序列, 该序列可为等间距或不等间距。运用这些插值点进行多维插值, 可以对不同输入变量采用线性或三次样条等不同插值法。

29.6 变速逻辑

汽车变速转换逻辑如图 29-6 所示。由图可知最上层的父状态模块为 `gear_selection`, `gear_state` 和 `selection_state` 为 `gear_selection` 两个并列的子状态模块。

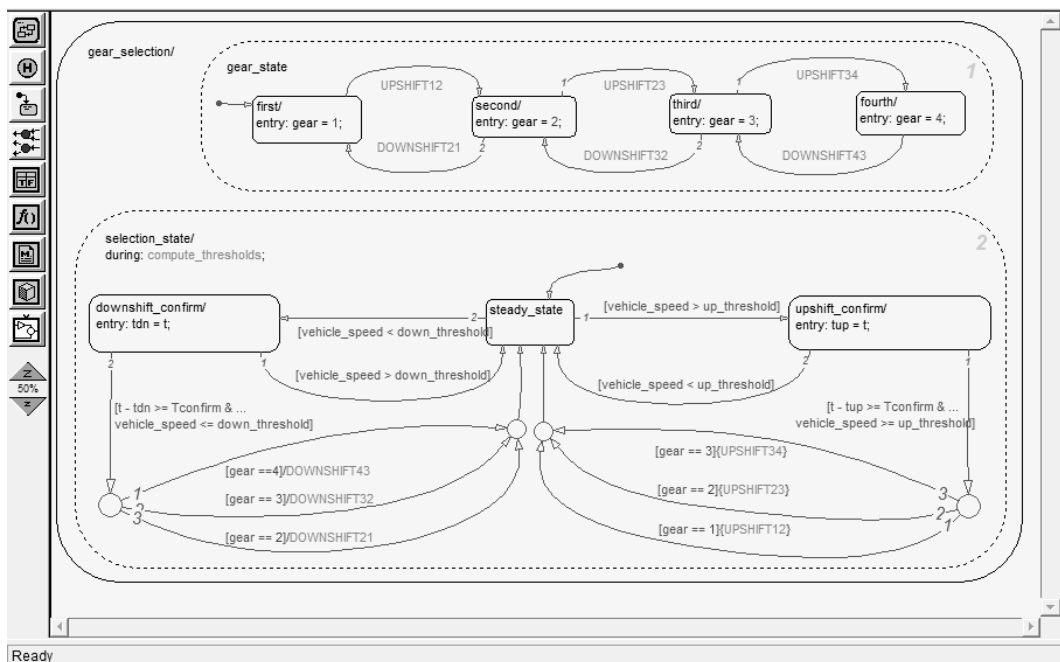


图 29-6 变速转换逻辑

`gear_state` 中包含四个独占模块: `first`、`second`、`third` 和 `fourth`, 分别表示变速器的四

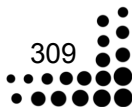


挡，状态模块之间通过升挡和降挡来实现转换。由于每一个模块最多与一个模块连接，因此一挡无法直接转移至三挡或者四挡。初始默认进入一挡。

`selection_state` 中包含三个独占模块：`downshift_confirm`、`steady_state` 和 `upshift_confirm`，默认进入稳定状态（`steady_state`），根据速度变化等条件进入与其他两个状态模块进行转移切换。

29.7 本例小结

本例以汽车为对象，对汽车传动系统进行了建模与仿真。汽车传动系统相对较为复杂，建模中利用到了 MATLAB 中多个技术，通过本例学习不仅可进一步加深对 Stateflow 的理解，还可对 MATLAB 中多个工具箱间配合仿真技术掌握更加牢靠。





第 30 例 导弹制导系统仿真

空空导弹在中远程精确打击、夺取制空权和空中对抗中发挥重要作用，它已成为世界许多国家优先发展和亮相购买的武器装备，但其结构复杂，试验成本过高，因此数学仿真必不可少。

空空导弹有导引系统、飞行控制系统、引战系统、推进系统、能源系统和弹体系统组成，其中导引系统和飞行控制系统是模型重点仿真部分。从导弹总体仿真，基于 Simulink 建立三自由度导弹数学模型，对工程实际有指导作用。

图 30-1 给出了导弹的外形图及参数。本例以导弹为对象，研究其三自由度动力学系统及仿真，通过本例学习，需掌握以下两部分内容：

- 熟练掌握 Stateflow 的使用方法。
- 导弹制导系统结构及组成。

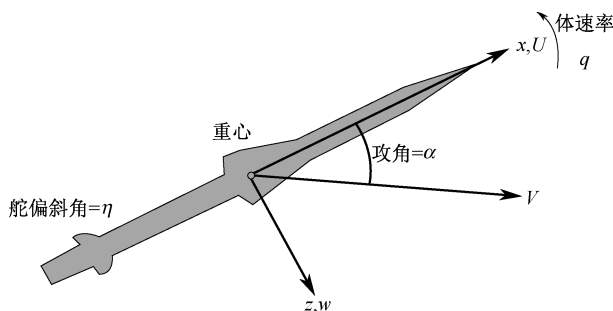


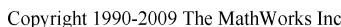
图 30-1 导弹外形图及参数

30.1 导弹三自由度动力学

导弹制导系统模型如图 30-2 所示，动力学模型如图 30-3 所示。本模型主要考虑导弹在铅垂面内的俯仰纵向运动，对于气动外形轴对称导弹，偏航通道和俯仰通道基本相同，原型设计主要基于导弹运动方程组。它主要包含描述导弹质心运动和弹体姿态变化的动力学方程、运动学方程、角度几何关系和描述控制系统工作方程。

操作关系方程的目标是按照导弹命中目标的要求，改变导弹速度和飞行的方向，即控制飞行。

由图 30-3 可以看出动力学模型分为三个部分：三自由度动力学、大气模型和自动驾驶仪，下面针对这三方面内容分别予以介绍。



The diagram illustrates the aircraft control system architecture, showing the interaction between the Atmosphere, Sensors, and Autopilot blocks.

Atmosphere Block: Provides environmental data to the Autopilot:

- Temperature(K)
- Speed of Sound(m/s)
- Air Pressure(N/m²)
- Air Density(Kg/m³)
- Height(m)

Sensors Block: Measures flight parameters and provides attitude data:

- q (roll rate)
- qdot (pitch rate)
- Ax, Az (accelerations)
- Az_m (measured angle of attack)
- Attitude (Xe, Ze)
- q (roll angle)

Autopilot Block: Processes sensor data and atmospheric information to generate control signals:

- Alpha (angle of attack)
- Mach (Mach number)
- Az_m (desired angle of attack)
- q_m (desired roll rate)
- Az_d (desired angle of attack derivative)
- Fin Demand (control signal)

Other Components:

- Incident (Incid):** Receives Alpha and Mach signals.
- Mach:** Receives Mach and Az_m signals.
- Miss_pos:** Receives a signal from the Autopilot and provides a reference for the Sensors.
- 1, 2, 3:** Reference signals for the Sensors.

图 30-3 导弹动力学模型

30.1.1 三自由度导弹动力学

假设导弹质量不变，无滚转角和滚转速度，无偏航角和偏航角速度，此时动力学模型为：



$$\dot{U} + QW = \sum F_{B_x} / m$$

$$\dot{W} - QU = \sum F_{B_z} / m$$

$$\dot{Q} = \sum M_Y / I_Y$$

$$\dot{\theta} = Q$$

$$\dot{X} = U \cos \theta + W \sin \theta$$

$$\dot{Z} = U \sin \theta - W \cos \theta$$

Z 轴向下为正，相对于重心的力和力矩如下：

$$\sum F_{B_x} = F_A - mg \sin \theta$$

$$\sum F_{B_z} = F_N + mg \cos \theta$$

$$\sum M_Y = M$$

轴向力、法向力和俯仰动量如下：

$$F_A = \frac{1}{2} \rho V^2 S C_A$$

$$F_N = \frac{1}{2} \rho V^2 S C_N$$

$$M = \frac{1}{2} \rho V^2 S d C_m$$

式中：

$$C_N = a_n \alpha^3 + b_n \alpha |\alpha| + C_n \left(2 - \frac{M}{3} \right) \alpha + d_n \delta$$

$$C_m = a_m \alpha^3 + b_m \alpha |\alpha| + C_m \left(-7 + \frac{8M}{3} \right) \alpha + d_m \delta + e_m Q$$

$$C_A = a_a$$

部分参数如下：

$$a_n = 19.373, a_m = 40.440, b_n = -31.023, b_m = -64.015$$

$$c_n = -9.717, c_m = 2.922, d_n = -1.948, d_m = -11.803$$

$$a_a = 300, e_m = -1.719$$

执行机构动力学：

$$\begin{bmatrix} \dot{\delta} \\ \ddot{\delta} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\omega_a^2 & -2\xi\omega_a \end{bmatrix} \begin{bmatrix} \delta \\ \dot{\delta} \end{bmatrix} + \begin{bmatrix} 0 \\ \omega_a^2 \end{bmatrix} \delta_c$$

式中 $\xi = 0.7$ ， $\omega_a = 150$ 。

轴向加速度： $n_x = F_A / gm - \sin \theta$

法向加速度： $n_z = F_N / gm + \cos \theta$

目标模块输出目标位置信息，然后和导弹位置求差得到弹目在地面坐标系中的坐标差，通过极坐标转换得到弹目距离和弹目视角；查找/跟踪模块以导弹俯仰姿态角、弹目距离和视角、预设的弹轴与目标视线夹角为输入，输出弹目距离、弹目接近速度、弹目视角角速度；导引处理模块主要输出控制的法向加速度和预设弹轴与目标视线的夹角；



自动驾驶仪及弹体模块以法向加速度为输入，以弹体的位置、俯仰姿态为输出，导弹与目标的位置形成模型最大回路，其中还有弹目视角和导弹姿态角差值回路等，以闭环增加了模型的稳定性。

30.1.2 大气模型

大气子系统所使用的是一个近似的国际标准的大气模型，并被分成两个单独的区域。在高于海平面 11 千米的对流层区域，在这个区域温度下降假定为随高度线性变化。第二区域在对流层 11 ~ 20 千米之间的平流层下部区域。假定在该区域中的温度保持不变。

空气动力学方程的运动子系统产生力和力矩，并集成了适用于体轴的导弹运动方程定义的线性和角运动的机身。

标准大气模型如下：

$$\rho(h) = \begin{cases} \rho_{osl} \exp^{-K_{\rho sl} h} & 0 & h & 36000 ft \\ \rho_{otr} \exp^{-K_{\rho tr}(h-36000)} & 36000 & h & 66000 ft \end{cases}$$

$$\rho(h) = \begin{cases} a_{sl} - K_{ah} & 0 & h & 36000 ft \\ a_{tr} & 36000 & h & 66000 ft \end{cases}$$

式中：

$$\rho_{osl} = 2.377 \times 10^{-3} lb \cdot s^2 / ft^2$$

$$\rho_{otr} = 0.7086 \times 10^{-3} lb \cdot s^2 / ft^2$$

$$K_{\rho sl} = 3.36174 \times 10^{-5} l / ft$$

$$K_{\rho tr} = 4.80377 \times 10^{-5} l / ft$$

$$a_{sl} = 1116.4 ft / s$$

$$a_{tr} = 968.1 ft / s$$

$$K_a = 0.00410833 l / ft$$

假设 α ——攻角， M ——马赫， Q ——倾斜角， γ ——飞行路径角， Z ——飞行高度， X ——水平距离，则有：

$$\tan \alpha = W / U, \quad V^2 = U^2 + W^2, \quad M = V / a, \quad \gamma = \theta - \alpha$$

$$\dot{M} = \frac{\rho a M^2 S}{2m} [C_A \cos \alpha + C_N \sin \alpha] - \frac{g}{a} \sin \gamma$$

$$\dot{\alpha} = \frac{\rho a M S}{2m} [C_N \cos \alpha - C_A \sin \alpha] + \frac{g}{a M} \cos \gamma + Q$$

$$\dot{\gamma} = -\frac{\rho a M S}{2m} [C_N \cos \alpha - C_A \sin \alpha] - \frac{g}{a M} \cos \gamma$$

$$\dot{Q} = \frac{\rho a^2 M^2 S d}{2I_y} C_m$$

$$\dot{Z} = M_a \sin \gamma$$

$$\dot{X} = M_a \cos \gamma$$



其中的一些物理常量如下：

I_y 为惯量，取值 $182.5 \text{ slug} \cdot \text{ft}^2$ 。

S 为参考面积，取值 0.44 ft^2 。

d 为参考距离，取值 0.75 ft 。

m 为质量，取值 13.98 slug 。

g 为重力，取值 32.2 ft/s^2 。

30.1.3 自动驾驶仪模型

自动驾驶仪是导弹制导与控制系统的重要组成部分，与导弹弹体构成的闭合回路称为稳定控制系统或稳定回路，自动驾驶仪的功能是控制及稳定导弹的飞行。即自动驾驶仪按控制指令的要求操纵舵面偏转或改变推力矢量方向，改变导弹的姿态，使导弹沿基准弹道飞行。

在稳定控制系统中，自动驾驶仪是控制器，导弹弹体是受控对象。自动驾驶仪的作用是稳定导弹质心的姿态运动，并根据引导指令正确、快速地控制导弹的飞行。

自动驾驶仪根据传入的法向加速度，按照闭环控制规律操纵升降舵以达到改变导弹姿态。在真实环境下，自动驾驶仪工作量非线性，因此模型使用二维 lookup 表实现自动驾驶仪的线性化，其依据是攻角与导弹马赫数。

30.2 导弹制导系统

制导系统和制导率如图 30-4、图 30-5 所示。

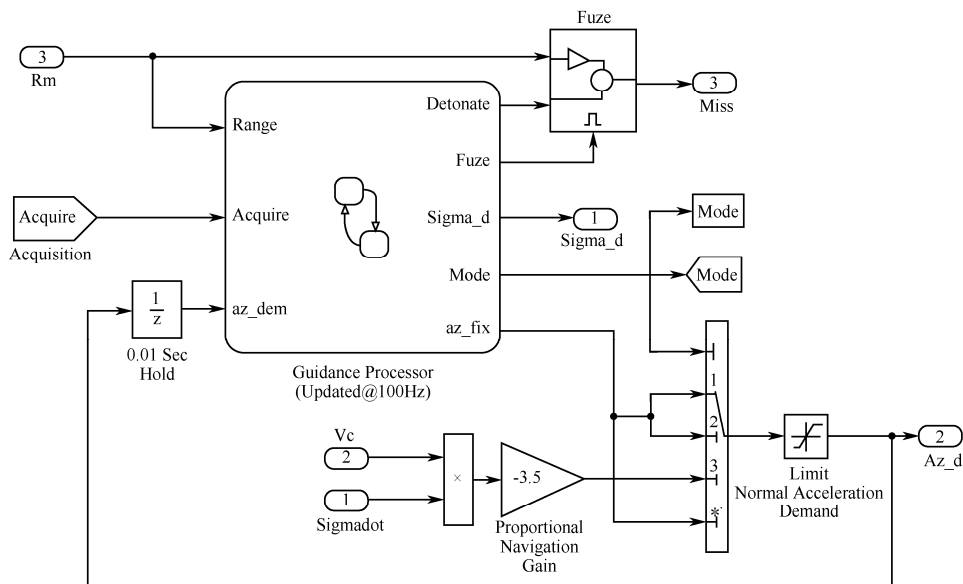


图 30-4 制导系统结构图

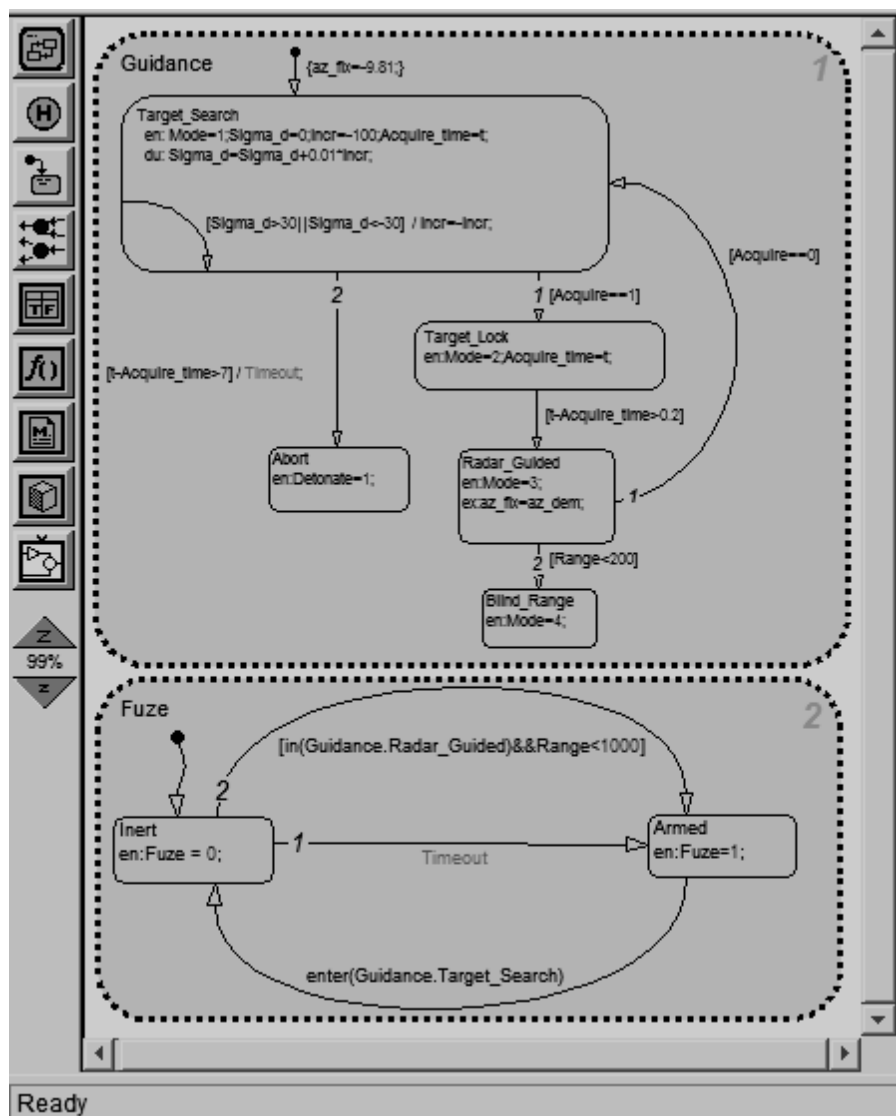


图 30-5 制导率

导引头原型分为两大部分，查找跟踪模块和导引处理模块。

1. 查找跟踪模块

查找跟踪模块主要根据几何关系提供目标是否捕获以及弹目视线角速度，导引处理功能之一是根据导引规则，利用弹目视线角速度提供法向加速度，导引处理模块还发出搜索指令以及在导弹不同的状态间切换。

图中每个方框代表导弹一种模式，有目标搜索、目标锁定、跟踪、引信触发四种模式。

2. 比例导引

在导引过程中的跟踪模式下，即在导弹捕获并锁定了目标后，该原型采用比例导引法。



比例导引法要求导弹在飞行过程中,保持速度矢量的转动角速度与目标视线的转动角速度成给定的比例关系。

$$\dot{\theta} = k \dot{\theta}_s$$

其中 k 为导引系数。

30.3 目标动力学

这里采用的目标动力学是假设目标固定于空间某点。目标动力学系统如图 30-6 所示。

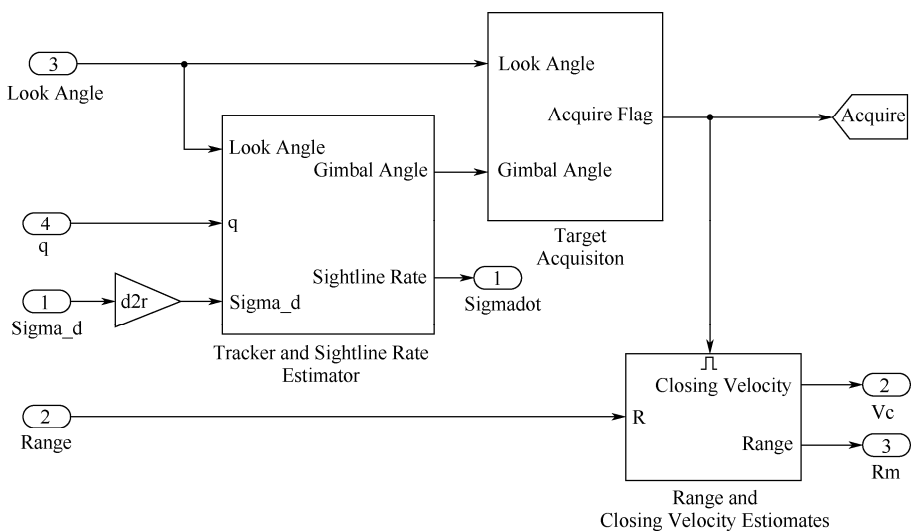


图 30-6 目标动力学

30.4 仿真结果

仿真结果如图 30-7 和图 30-8 所示。

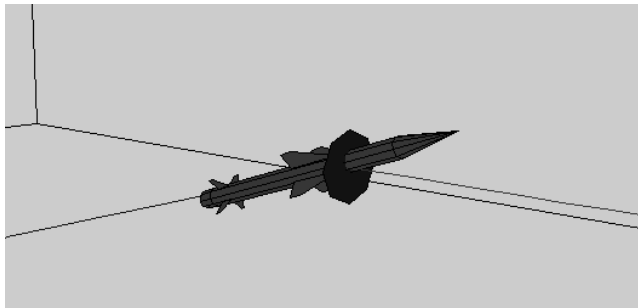


图 30-7 导弹击中目标

导弹曲线对于三自由度模型，导弹只在铅垂面内俯仰运动。假设目标在前上方以恒定速度相向飞来。导弹首先直线飞行，同时开始搜索目标，当获取锁定目标后，开始进入跟踪模式，按照导引律以一定的曲线弹道向目标飞行，直至最终命中目标。

由图 30-8 可见，在仿真起始一段时间内，导弹处于搜索目标状态，因此攻角为 0 或很小，当导引头发现目标下，攻角变大，跟踪目标稳定以后攻角逐渐减小，命中目标时接近 0。

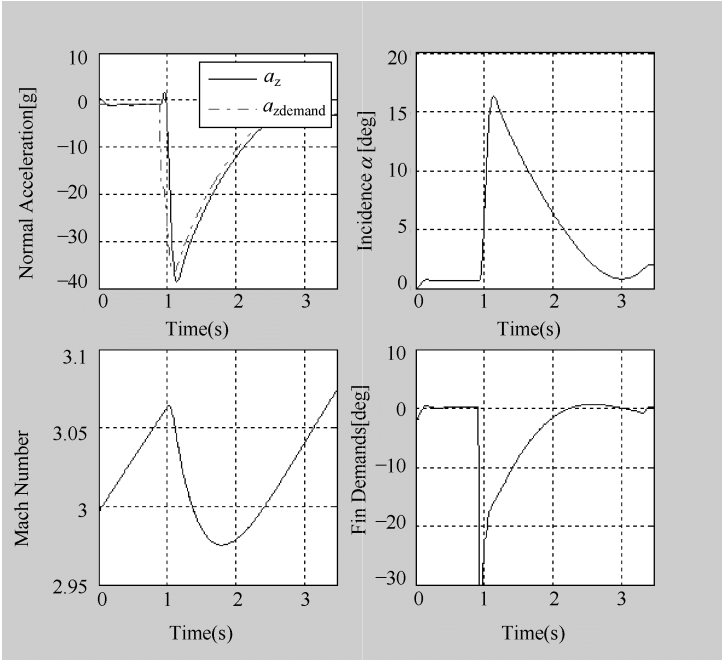


图 30-8 导弹参数变化

30.5 本例小结

本例以导弹为对象，对三自由度导弹制导控制进行了建模与仿真。通过本例学习进一步加深对 Stateflow 的理解，并对刚体动力学及制导律有所了解。

读者调查及投稿

1. 您觉得这本书怎么样？有什么不足？还能有什么改进？

2. 您在什么行业？从事什么工作？需要哪些方面的图书？

3. 您有无写作意向？愿意编写哪方面的图书？

4. 其他：

说明：

针对以上调查项目，可通过电子邮件直接联系：bjcwk@163.com 联系人：陈编辑

欢迎您的反馈和投稿！

电子工业出版社

反侵权盗版声明

电子工业出版社依法对本作品享有专有出版权。任何未经权利人书面许可，复制、销售或通过信息网络传播本作品的行为；歪曲、篡改、剽窃本作品的行为，均违反《中华人民共和国著作权法》，其行为人应承担相应的民事责任 and 行政责任，构成犯罪的，将被依法追究刑事责任。

为了维护市场秩序，保护权利人的合法权益，我社将依法查处和打击侵权盗版的单位和个人。欢迎社会各界人士积极举报侵权盗版行为，本社将奖励举报有功人员，并保证举报人的信息不被泄露。

举报电话：(010) 88254396; (010) 88258888

传 真：(010) 88254397

E-mail: dbqq@phei.com.cn

通信地址：北京市万寿路 173 信箱

电子工业出版社总编办公室

邮 编：100036